LEARNING STRUCTURED INFORMATION FROM LANGUAGE

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Arzoo Katiyar

August 2019

LEARNING STRUCTURED INFORMATION

FROM LANGUAGE

Arzoo Katiyar, Ph.D.

Cornell University 2019

Extracting information from text entails deriving a structured, and typically domain-specific, representation of entities and relations from unstructured text. The information thus extracted can potentially facilitate applications such as question answering, information retrieval, conversational dialogue and opinion analysis. However, extracting information from text in a structured form is difficult: it requires understanding words and the relations that exist between them in the context of both the current sentence and the document as a whole.

In this thesis, we present our research on neural models that learn structured output representations comprised of textual mentions of entities and relations within a sentence. In particular, we propose the use of novel output representations that allow the neural models to learn better dependencies in the output structure and achieve state-of-the-art performance on both tasks. We also propose models which can learn nested variation of the problem of entity mentions and achieves state-of-the-art performance. We also present our recent work on expanding the input context beyond sentences by incorporating coreference resolution to learn entity-level rather than mention-level representations and show that these representations are important for improving relation extraction. We perform analysis to show that the entity-level representations which capture the information regarding the saliency of entities in the document are beneficial for relation extraction. We also briefly mention about incorporating biases into the neural network models and show improvements in the performance of information extraction.

BIOGRAPHICAL SKETCH

Arzoo was born in Hapur, India but moved a lot with her parents, Usha and Jai and her brother, Aman before settling in Kanpur, India at the age of 7. She went to Dr. VSEC Awadhpuri for school where she developed interest in Physics because of her inspiring school teacher, Mrs. Vineeta Aggarwal. Later, she went to IIT Kanpur for undergraduate studies. She enjoyed algorithms the most and ended up taking many graduate level algorithms courses during her undergrad. However, towards the end in her undergraduate studies she started working on research projects in Data Mining and Databases and enjoyed them a lot because of their practical applications. Motivated by the thought of developing useful applications, she did her masters' thesis on efficient route planning in road networks which also got published as a research paper in a conference. She received several academic excellence award during her undergraduate studies. She also did internship at UBC Vancouver with Prof. David Poole in artificial intelligence. She greatly values the research experience in different areas in Computer Science during her undergraduate years.

After completion of the Bachelor's and Master's dual degree at IIT Kanpur, she joined Cornell University to pursue Phd in Computer Science. She enjoyed reading and discussing machine learning and natural language processing papers in various reading groups. She was intrigued by the neural network models that were just being introduced in the field of natural language processing when she started her Phd and later developed several neural network models for tasks in information extraction. She is still intrigued by the neural networks at the end of her Phd and plans to spend a lot of time understanding them. She also interned at IBM and MSR where she worked on biomedical domain which has inspired her to look for applications of natural language processing models in different areas such as healthcare. Apart from work, she likes to play all racquet sports, enjoys hiking and camping and is an occasional runner.

This document is dedicated to my parents, Usha and Jai and my brother, Aman.

ACKNOWLEDGEMENTS

I am extremely grateful to my advisor, Claire Cardie for her motivation, guidance, trust, support, patience, suggestions, time and help during my Phd. Her kindness and compassion made this journey easier manifold. I want to thank her for always being so excited and interested in all the different problems that I wanted to pursue during the past six years. I am also very thankful to my committee members Robert Kleinberg and David Mimno for their input and feedback on my research.

I would also like to thank my mentors during my internships at IBM Research, Ashish Jagmohan and MSR Redmond, Chris Quirk and Hoifung Poon, who inspired me and helped me gain a broader perspective about the impact of my research. I would also like to thank Bishan Yang and Ozan Irsoy for the insightful discussions during the start of my Phd which often showed me the path forward.

I also want to thank Vlad Niculae, Xilun Chen, Tianze Shi, Ashudeep Singh, Ana Smith, Jack Hessel, Moontae Lee, Laure Thompson, Xanda Schofield, Lu Wang, Jon Park, Rishi Bommasani, Tobias Schnabel, Adith Swaminathan, Karthik Raman, Xinya Du, Dipendra Misra, Lillian Lee, Yoav Artzi and all the members of the NLP seminar and mldg reading group for fun discussions and providing feedback on crazy ideas, paper drafts, etc.

I also want to thank my job material preparation team members Xanda Schofield, Theodoros Lykouris, Natacha Crooks and Jonathan Shi for their support and advice during the challenging process of preparation for academic job interviews. I learnt a lot about the preparation of applications during those meetings.

I also feel blessed to have known and spent time with my friends outside my discipline and have learnt a lot from them. Thank you Chaitali Joshi, Sophie Drame Maigne, Gauri Patwardhan, Ritika Dusad, Bola for being my roommates at different times of my Phd and for listening to my endless rants about work and life. Your company always

made the end of a challenging day more than bearable. I also want to thank Pankaj Singh, Abhishek Srivastava and Apoorva for making me feel comfortable in Ithaca during the initial days and inviting me over for dinner almost every week. I also had many dinners and a few memorable karaoke sessions with Ved Gund. Thank you for them. I also feel grateful for having friends who were always willing to accompany me to various activities such as running, swimming, squash, tennis, table tennis, badminton – Chaitali Joshi, Neeraj Kulkarni, Nikita Senger, Pragya Shah, Ritika Dusad, Ashudeep Singh, Utkarsh Mall, Saksham Agarwal, Anuj Bhargava, Drishti Wali. Because of these friends, I am graduating healthier than when I joined! I also want to thank Shraddha Vartak who kept me close to India and the festivals back home by enthusiastically celebrating them. I have received so much love from everyone I know in Ithaca that it feels like home. My wingies from IIT Kanpur who have been friends for more than 7 years and now going strong for a lifetime of friendship, have stayed with me and supported me during this time.

The list of people who made Ithaca home will not be complete without mentioning Prateek. I feel so fortunate to have gained a friend like him early in my Phd because he has been constantly there for me for every important or unimportant event in my life since then. He is my constant sports partner, cooking partner, movie partner, travel partner, discussion partner. His constant curiosity and encouragement kept me sailing during this time. I never needed to look any further for borrowing excitement or enthusiasm about research and life. Prateek, I can never thank you enough as I look forward to your constant supply of motivation and enthusiasm in the future.

Last, I want to thank my family who stood beside me and gave me encouragements whenever I need it the most. I could not have reached here without their love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

The vast amount of knowledge that is available on the web has inspired several approaches (Singh, 2018) in the field of information extraction to automatically extract information from it. This automatically extracted information can then be organized and presented to the users in a structured manner. As a result, this structured information is easier to understand and navigate[1] by the users in comparison to the plain text that encompasses this information. For example, search engines such as Google provide summaries as shown in Figure 1.1. In the summary shown in Figure 1.1, the search engine provides information about the person "Barack Obama" such as he was the 44th U.S. President, his birth date, his presidential term; his *relation* with places — he was born in "Honolulu"; his *relation* with other people — "Ann Dunham" and "Barack Obama Sr." are his parents and "Malia Ann Obama" and "Sasha Obama" are his children. It is also possible to navigate to these places and people from this summary. In addition to information organization, this automatically extracted knowledge is deeemed important for downstream tasks such as question answering (Chen, 2018). Consider, for example, the question shown in Figure 1.2. In order to build systems that can answer questions such as "Which U.S. president was born in Honolulu?" based on the information on the web, it is foreseeable that a system which has access to the information that the person "Barack Obama" served as the "U.S. President" and "Barack Obama" was also born in the city "Honolulu" and can perform inference on these *facts* can answer this question. Hence, extracting information can help in both organizing

[1] Easier navigation of information is possible due to the summarization of information from multiple sources on the web.

Figure 1.1: Figure showing an example of information organization for personality "Barack Obama" and the city "Honolulu".

information and building downstream systems such as opinion mining(Liu and Zhang, 2012), question answering (Chen, 2018), dialogue (Chen et al., 2017).

The automatic extraction of knowledge from web entails extracting real-world *entities* and their *relations* to each other. The semantic types of these entities and relations depends on the task that we are trying to solve in Natural Language Processing. For example, for general information extraction of extracting facts (Singh, 2018) these entities comprises of people, organization, locations, etc. which exist in the real-world and the relations between these entities comprises of the Organization-Affiliation (ORG-AFF) between a person and their organization, Physical (PHYS) between a person and their location, etc. For the task of extracting opinions (Liu and Zhang, 2012) that are expressed on the web, the entities correspond to opinion expression, the holder of the opinion and the target of the opinion. The relation between these opinion entities captures which opinion was expressed by which entities in the text. There are other tasks such as semantic role labeling (Màrquez et al., 2008), events extraction (Hogenboom et al., 2016), etc. In this thesis, we will primarily focus on extracting facts and opinions from text using machine learning algorithms.

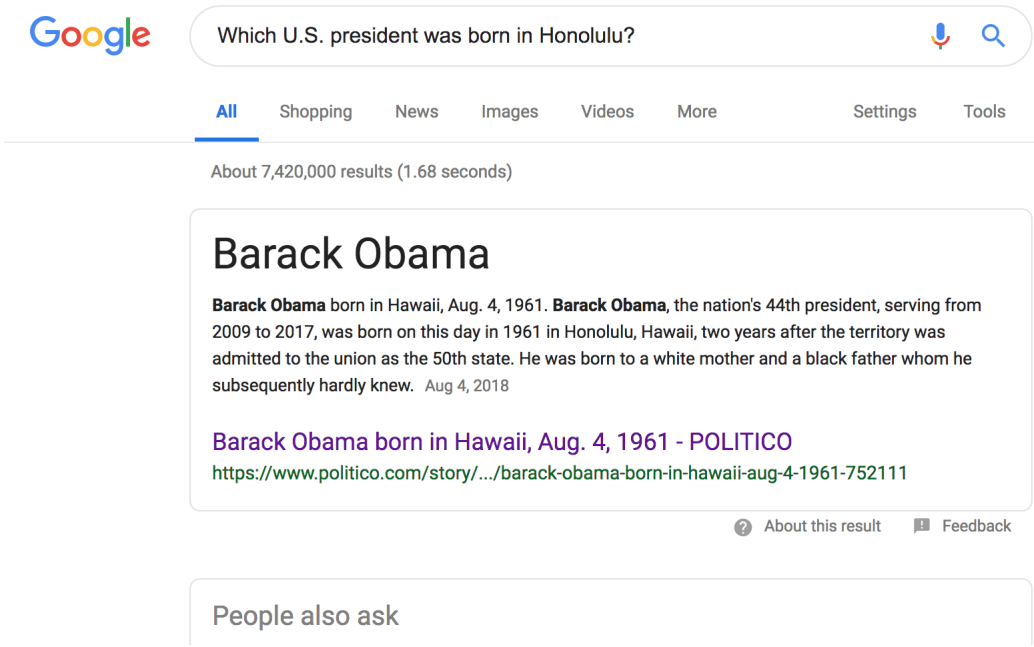


Figure 1.2: Figure showing an example of question answering from the popular search engine Google.

In all the tasks described above, the key assumption is that the entities are described in the text as a span of text, also referred to as mentions of entities. Hence, the general task of entities and relation extraction requires us to extract the textual spans and their relations to each other. These tasks are challenging because they require natural language understanding. A lot of the cues about the entities are present in the neighbouring tokens. For example, humans are able to identify new real-world entities in a stand-alone document without requiring external knowledge or databases. There are syntactic and semantic cues in a sentence which can help identify entities. Similarly, identifying relations between the entities require a deeper understanding of the context which provides cues about the type of relationship between any two entities. For example, syntactic structure of the sentence has been found to be extremely important for relation classification (Pawar et al., 2017). Similarly, semantic frames (Ruppenhofer et al., 2006) were often used as cues for machine learning algorithms to identify relations between known

entities. However, the approaches which used these discrete features as cues suffer from the limitation that generating these features is expensive and time-consuming and often depend on other active research problems in Natural Language Processing which can lead to cascading errors. However, the introduction of neural networks particularly, recurrent neural networks (Hochreiter and Schmidhuber, 1997) enabled building models which can automatically extract features from context. Recurrent neural networks is a type of neural networks which has connections over a temporal sequence. For a sentence, this temporal sequence corresponds to words [2] in the sentence from left-to-right. These networks are initialized with dense embeddings (Mikolov et al., 2013) trained on a very large corpus and known to capture semantics of words. With this initialization, recurrent neural networks then updates its hidden states (of features) for each word in the sequences, as it goes through the words in the sequence from left-to-right. Thus, these networks capture summary of the sequence seen so far at any time step in the sequence. These hidden states or features arguably capture features such as the context which were often used in the previous feature-based methods. However, there is also some evidence (Kuncoro et al., 2017, 2018) that these features are also capable of capturing syntactic structures of sentences. Thus, these recurrent neural networks can potentially be trained to *automatically* extract features which have been proven to be useful for extracting facts and opinion in the past work. In this thesis, we will present one of the *first* work on investigating the use of neural networks for extracting opinions from text.

Another challenge in extracting facts from a sentence lies in the complexity of the output space. Even under the assumption that these entities and relations are confined within a sentence, the complexity of the output space is, understandably, exponential in the length of the sentence N. Assuming that there are K entity types and R relation

[2]There are also variations where the temporal sequence is at the granularity of characters in the sentence. However, for this thesis we will restrict ourselves at the word-level.

types. An approach that labels each word (or token) in the sentence with the respective entity and simultaneously also indicates the position of the token within the entity [3] has cK possibilities for each token. Thus the output space corresponding to all the chain structures becomes exponential ($\approx$ cK^N) in the length of the sentence. However, there are approaches (Collobert et al., 2011) which relax dependencies among the chain structures reducing the complexity to quadratic in the length of the sentence. But jointly representing relations between entity mentions brings additional complexity in the output space. Thus, a naive representation of the output structure for relations makes it challenging for the current machine learning algorithms to learn these structures efficiently. Thus, there is a focus to propose novel output representations for these tasks which makes joint learning and inference of these tasks feasible. In this thesis we will show that novel output representations for extracting facts and opinions in the text, in conjunction with automatically extracting features from text using recurrent neural networks achieves the state-of-the-art performance on these problems.

Going forward, even though in this thesis, we will show that these neural networks are effective in automatically extracting features and can improve the state-of-the-art performance on the task of entities and relation extraction, there are several challenges in the field of information extraction that need to be addressed. One such challenge is that the task of extracting facts is often limited to *sentence-level*. One disadvantage of such an approach is that the methods extract relations between the mentions of the entities which are often pronominal mentions hence are not directly useful for any application unless there is a external system which can link the mention to its corresponding named entity in the document. Also, building systems independently often lead to cascading errors and thus multi-task learning (Li and Ji, 2014) is often helpful and achieves better performance that solving each task individually. In the context of extracting facts such

[3] An entity spans multiple tokens in the sentence which requires indicating the first token of an entity, middle tokens of the entity and the last token of the entity.

as entity mentions and relations, the task of also identifying the named entity for each mention is important. In this thesis, we present approaches to also learn entity-centric features and show that these features are important for learning relations.

In summary, in this thesis, we address the challenges of extracting facts and opinions from text by showing that using neural networks and proposed novel output representations, we can improve the state-of-the-art performance in information extraction. We also propose methods to incorporate features from the document for extracting facts and overcoming the limitation of the current recurrent network systems on their effectiveness on long context.

1.2 Contributions

The primary contribution of this thesis is the development of new methods for extracting options and facts based on automatic feature extraction and novel ways to represent these problems. We also propose methods to extend the current models to include context from document, thus paving the path for information extraction from the document as a whole instead of traditional way of extracting from sentences. More specifically, we make the following contributions:

Novel output representation for Opinion Relations. We extract opinion entities and relations from sentences using a variation of recurrent neural network called long short term memory (LSTM). Our major contribution is to represent relations between opinion entities in the terms of the distance between the related opinion entities such that the desired output structure is a sequence. We show that recurrent neural network models can learn this new representation of relations and performs comparably to the previous

state-of-the-art model (Yang and Cardie, 2013) which depends on the availability of linguistic knowledge and requires several expensive preprocessing components. Our other contribution is to be able to learn the output sequence globally and show that our methods achieves significant improvement over our baselines that learn this sequence locally. We will discuss our approach (Katiyar and Cardie, 2016) in detail in Chapter 3.

Relation Extraction without using Dependency Trees. Dependency trees have often been used for relation extraction (Pawar et al., 2017). Some approaches (Miwa and Bansal, 2016b) that used neural networks to automatically extract features for relation extraction still depend on the availability of dependency trees for sentences and used recurrent networks on tree structures for identifying relations between entity mentions. These approaches are not transferable to low-resource languages and obtaining dependency trees from another system also potentially leads to cascading errors. Thus, we investigate an approach which does not use dependency trees for relation extraction. We describe our approach in Chapter 4. In our approach, we use pointer networks (Vinyals et al., 2015a) which learns to attend to the related entity mention from a set of candidate entity mentions, for every mention in the sentence. We show that our approach (Katiyar and Cardie, 2017) can perform comparably to the approach which depends on the dependency tree structures.

Nested Named Entity Recognition. Named entities and their mentions are often nested [4]. For example, in the entity mention "his fellow pilot", "his" is another embedded entity mention. For extracting these entity mentions of the same type, a sequential output structure is not sufficient. We will propose to use a hypergraph structure in Chapter 5 and show that our output representation can be successfully learned using LSTMs

[4] In datasets such as ACE05, 30% of the sentences were found to have overlapping mentions.

and achieve the state-of-the-art performance (Katiyar and Cardie, 2018).

Introducing document-level features for relation extraction. Traditionally, relation extraction, often restricted to sentence-level, deal with identifying relations between entity mentions. In the work described in Chapter 6, we show that incorporating document-level features is important for the task of relation extraction achieving state-of-the-art performance. In particular, we learn entity-level representations aggregating information from all mentions of an entity. These entity-level representations capture the summary of the entities as they appear in the document.

1.3 Thesis Outline

The rest of the thesis is organized as follows. In Chapter 2, we will provide background in structured prediction models and an overview of recurrent neural networks. In the next three chapters, we will discuss different problems in fact and opinion extraction and we will present novel output representations for solving these problems. In Chapter 3, we will discuss novel neural network based models for opinion mining. In Chapter 4, we will present models for general fact extraction for joint modeling of entities and relations without requiring any external syntactic structures. In Chapter 5, we will provide neural network based models for nested named entities extraction. In the remaining thesis, we will discuss how we can improve automatic feature extraction from neural networks to include document-level features in the task of relation extraction. In Chapter 6, we will propose learning entity-level representations for relation extraction.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, we first present a definition and background of information extraction. We then present an overview of the general machine learning approaches for information extraction. In particular, we will focus on the approaches for extracting mentions of entities which are spans of text and relations between the entities. We will first formalize the problem and then describe feature-based approaches and then more recent but preliminary neural network approaches for them. In the next chapters, we will discuss our proposed neural network models for these problems in detail.

2.1 Information Extraction

As per Okurowski (1993), Information extraction is defined as *a kind of document processing which captures and outputs factual information contained within the document.* This factual information is not necessarily a summary of the document but it corresponds to extracting information based on predefined types of interest by identifying the specific instances of these types as they occur in the text or document. For example, a user interested in identifying information on companies named within a set of documents also includes identifying companies that were previously not known to the user.

Even with manual extraction, Will (1993) found in carefully controlled experiments, that analysts had an error rate of about 30% after training and several months of practice. To perform this task successfully, the analysts first locate a candidate entity by identifying and distinguishing a single entity (**entity extraction**) from the other entities, generally on the basis of the entity name. This candidate entity must be kept distinct from the other entities, but, in addition, other references to the same entity (**coreference**

resolution) must be merged so that there is a single entity in the extraction template. Characteristics of the entity must be assigned; the analyst can characterize the entity by nationality or classify the entity by type as one of the set of choices (**entity classification**). In an application where relationships among entities are important, the analyst may need to link one entity to another (**relation extraction**). All of these activities make for a complex set of cognitive demands placed upon the analyst that often require subtle judgements to be made.

Based on the challenges described above, several automatic approaches (Singh, 2018) have been proposed to solve the different components for information extraction such as entity extraction, coreference resolution, relation classification either independently or jointly. We will now describe some of the early machine learning approaches for these problems.

2.2 Machine Learning Approaches

We will primarily describe supervised machine learning based approaches for entity and relation extraction. We will focus on the feature-based methods for these problems and then describe the primitive deep learning approaches for these problems. Our work described in the later chapters is built on improving deep learning approaches.

2.2.1 Entity Extraction

Entity extraction entails extracting spans of text from a document that correspond to real world entities. It also entails extracting mentions of the entities in the document. Most of the current approaches extract entity mentions independently from each sentence in

Input sequence (x)	The	sale	infuriated	Beijing	which	regards	Taiwan	an	integral	...
Entity tags (y)	*B_T*	*I_T*	*B_O*	*B_H*	O	*B_O*	*B_T*	O	O	...

Figure 2.1: Output sequence for extracting opinion entities in the sentence . O represents the 'Other' tag in the BIO scheme.

the document. We will now describe the problem formally in detail.

2.2.1.1 Problem Formulation

Given an observed sequence, say a sentence, $\mathbf{x} = \langle x_1, x_2, \ldots x_t \rangle$, with each $x_i \in V$, vocabulary of V words in language, we want to extract multiple spans $[p, p + l_1], [q, q + l_2] \ldots$ of variable lengths which correspond to mentions of entities in the sentence. These spans can be overlapping, however, for this chapter we will only discuss non-overlapping spans. Under the assumption that these entity mentions are non-overlapping, most successful (Ramshaw and Marcus, 1995) output representation to extract all spans of entity mentions correspond to a sequence $\mathbf{y} = \langle y_1, y_2, \ldots y_t \rangle$ such that there is a label y_k for each x_k in the input sequence. In this work, we follow BILOU tagging sequence where B corresponds to the beginning of the entity mention, I corresponds to the inside, L corresponds to the last token, O corresponds to the outside token which is not a part of any entity mention and U represents an entity mention comprising of single token. These tags are appended by the type of the entity mention to distinguish between the different types of entity mentions. Figure 2.1 shows an example of the output sequence (y) for extracting opinion entities "The sale", "infuriated", "Beijing", "regard", "Taiwan" from the sentence.

Previous methods (Collobert et al., 2011) use machine learning models to learn to predict the sequence y, given an input sequence x. We will describe a simple model that

predicts each y_i independently in the output sequence. [1] For methods to be generalizable over unseen input sequences, most of the previous methods (Nadeau and Sekine, 2007) use linguistic knowledge to represent the sentence using a feature representation for each prediction task. However, more recent neural network models have been shown to learn linguistic features by themselves. We will now describe these two class of models in detail.

2.2.1.2 Feature-based models

For each prediction task, i.e., to predict y_i given the sentence x, these methods extract linguistic features from the sentence such as

- **Words** which correspond to the current word x_i and the neighboring words in a window [i-2, i+2]
- **Capitalization features** for the current word x_i
- **Part-of-speech features** for the current word x_i and the neighboring words in a window [i-2, i+2]
- **Dependency tree features** for the current word x_i, determine its syntactic chunks, etc
- **Semantic class features** semantic frames for verbs according to FrameNet

These include some of the common features that are used by the feature-based systems (Nadeau and Sekine, 2007). For each specific problem such as identifying opinion entities, opinion lexicons are used which indicate the presence of an opinion. Also, the output from other NLP systems such as identifying the named entities are used as additional features for identifying opinion entities. Even some of the features described

[1] We will discuss how we can model dependencies in the output structure in Chapter 3.

earlier such as part-of-speech features, dependency trees and constituent trees are not easily available and hence obtaining these features also depend on other NLP systems.

Thus, the features described above are used to obtain a feature representation f_i for each word x_i in the input sequence. The machine learning based approaches learns weights for each of these features in the feature representation, to learn to predict the corresponding output tag y_i in the output sequence.

These machine learning approaches achieved modest performance on entity extraction and relation classification. However, there are several limitations of using these feature-based approaches such as:

- These methods depend on linguistic knowledge for building feature representation and as a result these methods depend on several current existing tools such as dependency parsers, constituency parsers, semantic role labeling and so on. These existing tools are far from perfect and most often domain-dependent which results in propagating errors in systems which uses these tools to extract features.
- Also, these methods depend on manually selecting the linguistic features for building feature representation which has the disadvantage of being limited by the expert understanding of which linguistic cues are important for a task and also it becomes increasingly difficult to know the linguistic signals important for solving complex downstream tasks.

2.2.2 Neural Network Approaches

In around 2015, neural networks became increasingly popular for entity extraction because of their capability to automatically extract features, models built on neural net-

works (Lample et al., 2016) improved the state-of-the-art performance in four languages without resorting to any language-specific knowledge or resources such as gazetteers. In this section, we will describe a simple sequence-to-sequence recurrent neural network based model architecture for entity extraction.

Learning Word Representations. We input spans of text to these neural networks. In our work, we consider words as the units of text. In the previous feature-based approaches, words are often represented as a "one-hot" vector of the size of the vocabulary which means that this vector has 1 corresponding to the index of the word in the vocabulary and all other indices are 0 (Manning and Schütze, 1999). These word representations do not capture any information about similarity or any other relationships between words. On the other hand, **distributed representations** aim to capture semantic similarity of words by embedding all words in a dense low-level representations comprising of real numbers. There have been many recent methods to learn these distributed representations from unsupervised data. Some of these methods (Collobert and Weston, 2008; Mikolov et al., 2013) use deep neural networks to learn these word representations which are known to show linear structures such as below:

$$queen \approx king - man + woman$$

where *queen*, *king*, *man* and *woman* are learned dense vector representations for the corresponding words.

These representations can be used as an input in the neural networks for tasks such as entity extraction and they have been found to provide a good initialization for these neural networks. We will now describe the recurrent neural networks to extract feature representations for entity extraction.

Recurrent neural networks. Recurrent neural networks are a type of neural networks which are known to capture sequentiality of the input sequence and hence well suited for tasks on natural language such as English where the natural order is from left to right. These networks employ non-linear operations on words from left-to-right such that they capture summary of the input seen so far at every timestep. Thus, these networks recursively apply a non-linear update function f on the history (h_{t-1}) and the current word (x_t) to produce h_t to capture the summary until timestep t as below:

$$h_t = f(h_{t-1}, x_t) \tag{2.1}$$

where h_t is the hidden representation at time t which encodes the information about the order of words in the sentence due to recurrent computation of h at every time step which is propagated further to the next time step. Thus, these hidden representations can capture context from the left. There are also variations of these recurrent neural networks such as a bidirectional neural network which can also capture the context from the right. Similarly, recurrent neural networks are often stacked together to obtain a multi-layer recurrent neural network. These hidden representations corresponding to each word in the input sequence are then used as feature representation for entity extraction.

Sequence Tagging. As we described earlier, we can obtain a hidden representation h_t for each time step t in the sentence which captures the context both to the left and right of the current timestep t using bidirectional multi-layer recurrent neural network. These hidden representations are then used as feature representations to learn to predict the most likely tag for each time step similar to Section 2.2.1.1. These models accumulate loss over each time step corresponding to all the words in the sentence and then are trained using *backpropagation* through time to compute the gradients and updating the parameters in the network. For our problem of entity extraction, the output sequence for

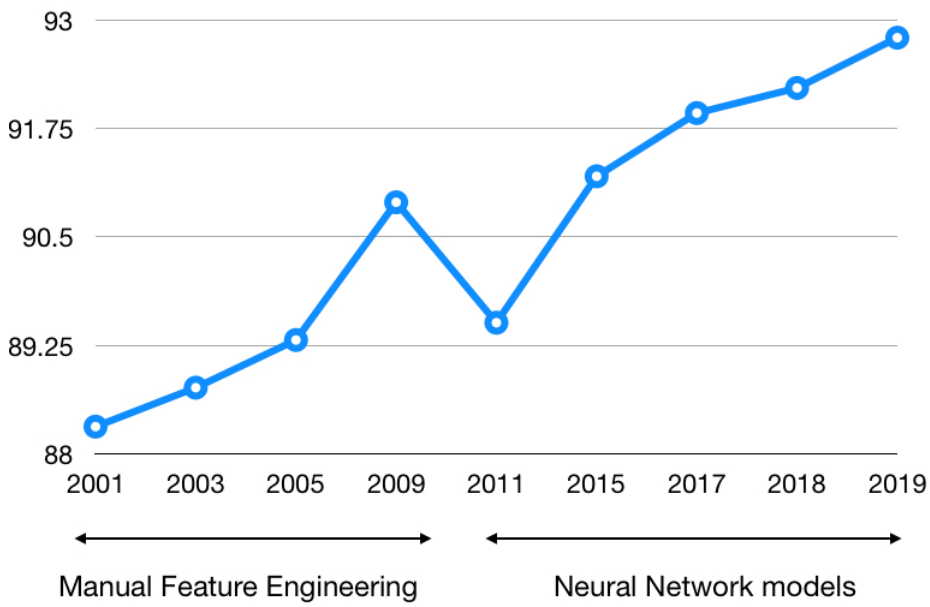


Figure 2.2: Trend in the F1-score performance of various feature-based and neural network based systems on CoNLL-2003 test set. Each blue dot in the graph corresponds to some of the recent model proposed in the literature.

each time step corresponds to the BILOU tag appended by the entity type.

These recurrent neural networks have been widely used in a variety of sequence tagging problems such as named entity recognition (Lample et al., 2016), semantic role labeling (He et al., 2017), etc. As shown in Figure 2.2, neural network models outperform previous feature-based models on the English CoNLL-2003 test set without any external resources. In addition to sequence tagging, recurrent neural networks have also been used for sequence-to-sequence problems such as machine translation () where an encoder embeds a sentence into a dense representation and then the decoder generates the target sentence from the representation. Bahdanau et al. (2015) proposed attention mechanism for learning the sentence representation by weighing the different parts of the encoder sequence by a learned probability distribution. These neural network methods have shown improvements in both the sequence-to-sequence as well as tagging tasks

without requiring manual feature-engineering to learn feature representation.

In this thesis, we focus on building neural network based models for information extraction including extracting opinion from text. We primarily want to build better neural networks because:

- The biggest advantage of using neural networks is that it does not depend on the output from an external NLP tasks to build the set of linguistic features but instead provides an opportunity for these features to be learned in an end-to-end fashion either using multi-task objectives or learning these features implicitly thus not leading to error propagation from other systems.
- Also, building a comprehensive list of linguistic features is both cumbersome and difficult, thus having neural networks learn these features also provide the advantage of being able to learn more complex predictive features not necessarily quantifiable by the experts for more challenging downstream tasks.

We will now cover some of the general approaches for relation extraction using both feature based models and few neural network models.

2.2.3 Relation Extraction

Relation Extraction deals with extracting relations of interest among mentions of entities in the text. In pipelined approaches entity mentions are extracted first and then relations between the entity mentions are extracted. The most common approach (Pawar et al., 2017) is to learn a binary (or multi-class) classifier between all pairs of entity mentions in the text to classify if the pair of entities are related (and their type of relation). In feature-based approaches (Yang and Cardie, 2013), most common features for relation

extraction include words, their POS tags, hypernyms, syntactic category in the parse tree, semantic frames, *dependency paths*, distance between the arguments (entity mentions) and so on. In despite of these extensive list of features required for performing relation extraction, Miwa and Bansal (2016b) show that they achieve error reduction of 12.1% on end-to-end relation extraction compared to the previous state-of-the-art (Li and Ji, 2014) feature-based model on ACE2005 dataset. Thus, neural network based models show promise for solving relation extraction. However, there are two major challenges that need to be addressed which are:

- Focusing on the development of **joint** approaches for entity and relation extraction because pipelined approaches lead to error propagation and any entity mention that was not discovered in the first step can't be extracted during relation extraction. Previous feature-based approaches (Li and Ji, 2014) show that extracting entities and relation jointly improves both the tasks. In Chapter 3, we will introduce our proposed approaches for joint opinion entities and relation extraction.
- Miwa and Bansal (2016b) use dependency trees from off-the-shelf parser to learn relation extraction as dependency paths have been known to be crucial for relation extraction. However, these parsers are not perfect which lead to error propagation. Thus, we need to build models which do not depend on the availability of parse trees. In Chapter 4, we investigate neural network models without using dependency trees.

In addition to the challenges described above, relation extraction is often restricted to find relation between entity mentions without unifying the mentions to their respective entities in the document. We will discuss our approach in Chapter 6 describing how to include entity-level information to relation extraction. We show that we can learn these entity-level representations using the coreference information to unify different

mentions of an entity and improve relation extraction. Thus, in this thesis we focus on including document-level information for extracting facts from document.

CHAPTER 3

FINE-GRAINED OPINION MINING

In this chapter, we present a deep learning based approach for fine-grained opinion mining. We propose a joint model for the extraction of opinion entities and relation by proposing a novel tagging scheme. The work described is based on Katiyar and Cardie (2016). We first present the definition of fine-grained opinion mining and the describe some of the previous approaches for this problem prior to the publication of our work. We then describe our model based on recurrent neural networks and present the performance of our approach compared to the previous state-of-the-art model for fine-grained opinion mining.

To give some background on the problem, the goal of fine-grained opinion analysis is to identify subjective expressions in text along with their associated sources and targets. More specifically, fine-grained opinion analysis aims to identify three types of *opinion entities*:

- **opinion expressions**, O, which are direct subjective expressions (i.e., explicit mentions of otherwise private states or speech events expressing private states (Wiebe and Cardie, 2005));
- **opinion targets**, T, which are the entities or topics that the opinion is about; and
- **opinion holders**, H, which are the entities expressing the opinion.

In addition, the task involves identifying the IS-FROM and IS-ABOUT relations between an opinion expression and its holder and target, respectively. In the sample sentences, numerical subscripts indicate an IS-FROM or IS-ABOUT relation.

S1 [The sale]$_{T_1}$ [infuriated]$_{O_1}$ [Beijing]$_{H_{1,2}}$ which [regards]$_{O_2}$ [Taiwan]$_{T_2}$ an integral part of its territory awaiting reunification, by force if necessary.

S2 "[Our agency]$_{T_1,H_2}$ [seriously needs]$_{O_2}$ [equipment for detecting drugs]$_{T_2}$," [he]$_{H_1}$ [said]$_{O_1}$.

In S1, for example, "infuriated" indicates that there is an (negative) opinion from "Beijing" regarding "the sale."[1]

Traditionally, the task of extracting opinion entities and opinion relations was handled in a pipelined manner, i.e., extracting the opinion expressions first and then extracting opinion targets and opinion holders based on their syntactic and semantic associations with the opinion expressions (Kim and Hovy, 2006; Kobayashi et al., 2007). More recently, methods that *jointly* infer the opinion entity and relation extraction tasks (e.g., using Integer Linear Programming (ILP)) have been introduced (Choi et al., 2006; Yang and Cardie, 2013) and show that the existence of opinion relations provides clues for the identification of opinion entities and vice-versa, and thus results in better performance than a pipelined approach. However, the success of these methods depends critically on the availability of opinion lexicons, dependency parsers, named-entity taggers, etc.

Alternatively, neural network-based methods have been employed. In these approaches, the required latent features are automatically learned as dense vectors of the hidden layers. Liu et al. (2015), for example, compare several variations of recurrent neural network methods and find that long short-term memory networks (LSTMs) perform the best in identifying opinion expressions and opinion targets for the specific case of product/service reviews.

Motivated by the recent success of LSTMs on this and other problems in NLP, we

[1]The work described in this chapter does not attempt to determine the sentiment, i.e., the positive or negative polarity, of an opinion.

investigate here the use of deep bi-directional LSTMs for joint extraction of opinion expressions, holders, targets and the relations that connect them. This is the first attempt to handle the full opinion entity and relation extraction task using a deep learning approach.

In experiments on the MPQA dataset for opinion entities (Wiebe and Cardie, 2005; Wilson, 2008), we find that standard LSTMs are not competitive with the state-of-the-art CRF+ILP joint inference approach of Yang and Cardie (2013), performing below even the standalone sequence-tagging CRF. Inspired by Huang et al. (2015), we show that incorporating sentence-level, and our newly proposed relation-level optimization, allows the LSTM to perform within 1–3% of the ILP joint model for all three opinion entity types and to do so without access to opinion lexicons, parsers or other preprocessing components.

For the primary task of identifying opinion entities together with their IS-FROM and IS-ABOUT relations, we show that the LSTM with sentence- and relation-level optimizations outperforms an LSTM baseline that does not employ joint inference. When compared to the CRF+ILP-based joint inference approach, the optimized LSTM performs slightly better for the IS-ABOUT[2] relation and within 3% for the IS-FROM relation.

In the sections that follow, we describe: related work (Section 4.1) and the multi-layer bi-directional LSTM (Section 3.2); the LSTM extensions (Section 6.3); the experiments on the MPQA corpus (Sections 6.4 and 6.5) and error analysis (Section 5.6).

[2]Target and IS-ABOUT relation identification is one important aspect of opinion analysis that hasn't been much addressed in previous work and has proven to be difficult for existing methods.

3.1 Related Work

LSTM-RNNs (Hochreiter and Schmidhuber, 1997) have recently been applied to many sequential modeling and prediction tasks, such as machine translation (Bahdanau et al., 2015; Sutskever et al., 2014), speech recognition (Graves et al., 2013), NER (Hammerton, 2003). The bi-directional variant of RNNs has been found to perform better as it incorporates information from both the past and the future (Schuster and Paliwal, 1997; Graves et al., 2013). Deep RNNs (stacked RNNs) (Schmidhuber, 1992; Hihi and Bengio, 1996) capture more abstract and higher-level representation in different layers and benefit sequence modeling tasks (İrsoy and Cardie, 2014). Collobert et al. (2011) found that adding dependencies between the tags in the output layer improves the performance of Semantic Role Labeling task. Later, Huang et al. (2015) also found that adding a CRF layer on top of bi-directional LSTMs to capture these dependencies can produce state-of-the-art performance on part-of-speech (POS), chunking and NER.

For fine-grained opinion extraction, earlier work (Wilson et al., 2005; Breck et al., 2007; Yang and Cardie, 2012) focused on extracting subjective phrases using a CRF-based approach from open-domain text such as news articles. Choi et al. (2005) extended the task to jointly extract opinion holders and these subjective expressions. Yang and Cardie (2013) proposed a ILP-based joint-inference model to jointly extract the opinion entities and opinion relations, which performed better than the pipelined based approaches (Kim and Hovy, 2006).

In the neural network domain, İrsoy and Cardie (2014) proposed a deep bi-directional recurrent neural network for identifying subjective expressions, outperforming the previous CRF-based models. Irsoy and Cardie (2013) additionally proposed a bi-directional recursive neural network over a binary parse tree to jointly identify opin-

ion entities, but performed significantly worse than the feature-rich CRF+ILP approach of Yang and Cardie (2013). Liu et al. (2015) used several variants of recurrent neural networks for joint opinion expression and aspect/target identification on customer reviews for restaurants and laptops, outperforming the feature-rich CRF based baseline. In the product reviews domain, however, the opinion holder is generally the reviewer and the task does not involve identification of relations between opinion entities. Hence, standard LSTMs are applicable in this domain. None of the above neural network based models can jointly model opinion entities and opinion relations.

In the relation extraction domain, several neural networks have been proposed for relation classification, such as RNN-based models (Socher et al., 2012) and LSTM-based models (Xu et al., 2015b). These models depend on constituent or dependency tree structures for relation classification, and also do not model entities jointly. Recently, Miwa and Bansal (2016b) proposed a model to jointly represent both entities and relations with shared parameters, but it is not a joint-inference framework.

3.2 Methodology

For our task, we propose the use of multi-layer bi-directional LSTMs, a type of recurrent neural network. Recurrent neural networks have recently been used for modeling sequential tasks. They are capable of modeling sequences of arbitrary length by repetitive application of a recurrent unit along the tokens in the sequence. However, recurrent neural networks are known to have several disadvantages like the problem of vanishing and exploding gradients. Because of these problems, it has been found that recurrent neural networks are not sufficient for modeling long term dependencies. Hochreiter and Schmidhuber (1997), thus proposed long short term memory (LSTMs), a variant of

recurrent neural networks.

3.2.1 Long Short Term Memory (LSTM)

Long short term memory networks are capable of learning long-term dependencies. The recurrent unit is replaced by a memory block. The memory block contains two cell states – memory cell C_t and hidden state h_t; and three multiplicative gates – input gate i_t, forget gate f_t and output gate o_t. These gates regulate the addition or removal of information to the cell state thus overcoming vanishing and exploding gradients as described below:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.1}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3.2}$$

The forget gate f_t and input gate i_t above decides what part of the information we are going to throw away from the cell state and what new information we are going to store in the cell state. The sigmoid outputs a number between 0 and 1 where 0 implies that the information is completely lost and 1 means that the information is completely retained.

The intermediate cell state $\widetilde{C}_t$ and previous cell state C_{t-1} are used to update the new cell state C_t as shown below:

$$\widetilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{3.3}$$

$$C_t = i_t * \widetilde{C}_t + f_t * C_{t-1} \tag{3.4}$$

Next, we update the hidden state h_t based on the output gate o_t and the cell state C_t. We pass both the cell state C_t and the hidden state h_t to the next time step as shown below:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o C_t + b_o) \tag{3.5}$$

$$h_t = o_t * \tanh(C_t) \tag{3.6}$$

3.2.2 Multi-layer Bi-directional LSTM

In sequence tagging problems, it has been found that only using past information for computing the hidden state h_t may not be sufficient. Hence, previous works (Graves et al., 2013; İrsoy and Cardie, 2014) proposed the use of bi-directional recurrent neural networks for speech and NLP tasks, respectively. The idea is to also process the sequence in the backward direction. Hence, we can compute the hidden state $\overrightarrow{h_t}$ in the forward direction and $\overleftarrow{h_t}$ in the backward direction for every token.

Also, in more traditional feed-forward networks, deep networks have been found to learn abstract and hierarchical representations of the input in different layers (Bengio, 2009). The multi-layer LSTMs have been proposed (Hermans and Schrauwen, 2013) to capture long-term dependencies of the input sequences in different layers.

For the first hidden layer, the computation proceeds similar to that described in Section 3.2.1. However, for higher hidden layers i the input to the memory block is the hidden state and memory cell from the previous layer $i - 1$ instead of the input vector representation.

For the work described in this chapter, we only use the hidden state from the last layer L to compute the output state y_t.

$$z_t = \overrightarrow{V h_t}^{(L)} + \overleftarrow{V h_t}^{(L)} + c \tag{3.7}$$

$$y_t = g(z_t) \tag{3.8}$$

3.3 Network Training

For our problem, we wish to predict a label y from a discrete set of classes Y for every word in a sentence. As is the norm, we train the network by maximizing the log-likelihood

$$\sum_{(x,y)\in \mathrm{T}} \log p(y|x,\theta)$$

over the training data T, with respect to the parameters θ, where x is the input sentence and y is the corresponding tag sequence. We propose three alternatives for the log-likelihood computation.

3.3.1 Word-Level Log-Likelihood (WLL)

We first formulate a word-level log-likelihood (WLL) (adapted from Collobert et al. (2011)) that considers all words in a sentence independently. We interpret the score z_t corresponding to the i^{th} tag $[z_t]_i$ as a conditional tag probability $\log p(i|x,\theta)$ by applying a softmax operation.

$$p(i|x,\theta) = softmax(z_t^i) \tag{3.9}$$

$$= \frac{e^{z_t^i}}{\sum_j e^{z_t^j}} \tag{3.10}$$

For the tag sequence y given the input sentence x the log-likelihood is :

$$\log p(y|x,\theta) = z^y - \operatorname*{logadd}_j z^j \tag{3.11}$$

3.3.2 Sentence-Level Log-Likelihood (SLL)

In the word-level approach above, we discard the dependencies between the tags in a tag sequence. In our sentence-level log-likelihood (SLL) formulation (also adapted from Collobert et al. (2011)) we incorporate these dependencies: we introduce a transition score $[A]_{i,j}$ for jumping from tag i to tag j of adjacent words in the tag sequence to the set of parameters $\widetilde{\theta}$. These transition scores are going to be trained.

We use both the transition scores $[A]$ and the output scores z to compute the sentence score $s(x|_{t=1}^{T}, y|_{t=1}^{T}, \widetilde{\theta})$ as below:

$$s(x, y, \widetilde{\theta}) = \sum_{t=1}^{T} \left([A]_{y_{t-1}, y_t} + z_t^{y_t} \right) \tag{3.12}$$

We normalize this sentence score over all possible paths of tag sequences $\widetilde{y}$ to get the log conditional probability as below :

$$\log p_{sent}(y|x, \widetilde{\theta}) = s(x, y, \widetilde{\theta}) - \underset{\widetilde{y}}{\text{logadd}}\, s(x, \widetilde{y}, \widetilde{\theta}) \tag{3.13}$$

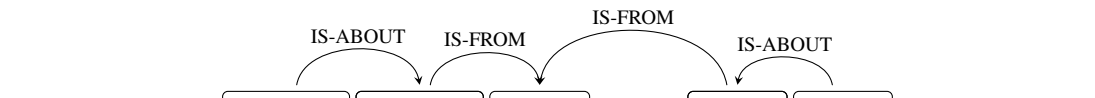
Even though the number of tag sequences grows exponentially with the length of the sentence, we can compute the normalization factor in linear time (Collobert et al., 2011).

At inference time, we find the best tag sequence

$$\underset{\widetilde{y}}{\text{argmax}}\, s(x, \widetilde{y}, \widetilde{\theta})$$

for an input sentence x using Viterbi decoding. In this case, we basically maximize the same likelihood as in a CRF except that a CRF is a linear model.

The above sentence-level log-likelihood is useful for sequential tagging, but it cannot be directly used for modeling relations between non-adjacent words in the sentence. In the next subsection, we extend the above idea to also model relations between non-adjacent words.



	The	sale	infuriated	Beijing	which	regards	Taiwan	an	integral	...
Entity tags	*B_T*	*I_T*	*B_O*	*B_H*	O	*B_O*	*B_T*	O	O	...
Left_Rel (d_{left})	0	0	0	0	0	2	1	0	0	...
Right_Rel (d_{right})	2	1	1	0	0	0	0	0	0	...

Figure 3.1: Gold standard annotation for an example sentence from MPQA dataset. O represents the 'Other' tag in the BIO scheme.

3.3.3 Relation-Level Log-Likelihood (RLL)

For every word x_t in the sentence x, we output the tag y_t and a distance d_t. If a word at position t is related to a word at position k and $k < t$, then $d_t = (t - k)$. If word t is not related to any other word to its left, then $d_t = 0$. Let D_{Left} be the maximum distance we model for such *left*-relations [3].

$$z_t = \overrightarrow{V_r}\overrightarrow{h_t}^{(L)} + \overleftarrow{V_r}\overleftarrow{h_t}^{(L)} + c_r \tag{3.14}$$

We let $\overrightarrow{V_r} \in \mathbb{R}^{(D_{Left}+1)\times Y \times d_h}$ (where d_h is the dimensionality of hidden units) such that the output state $z_t \in \mathbb{R}^{(D_{Left}+1)\times Y}$ as compared to $z_t \in \mathbb{R}^{(1)\times Y}$ in case of sentence-level log-likelihood.

In order to add dependencies between tags and relations, we introduce a transition score $[A]_{i,j,d',d"}$ for jumping from tag i and relation distance d' to tag j and relation distance $d"$ of adjacent words in the tag sequence, to the set of parameters θ'. These transition scores are also going to be trained similar to the transition scores in sentence-level log-likelihood.

[3] Later in this section, we will also add a similar likelihood in the objective function for *right*-relations, i.e., for each word the related words are in the right context.

The sentence score $s(x|_{t=1}^{T}, y|_{t=1}^{T}, d|_{t=1}^{T}, \theta')$ is:

$$s(x, y, d, \theta') = \sum_{t=1}^{T} \left([A]_{y_{t-1}, y_t, d_{t-1}, d_t} + z_t^{y_t, d_t} \right) \tag{3.15}$$

We normalize this sentence score over all possible paths of tag $\widetilde{y}$ and relation sequences $\widetilde{d}$ to get the log conditional probability as below :

$$\log p_{rel,Left}(y, d|x, \widetilde{\theta}) = s(x, y, d, \theta') \tag{3.16}$$

$$- \operatorname*{logadd}_{\widetilde{y}, \widetilde{d}} s(x, \widetilde{y}, \widetilde{d}, \theta') \tag{3.17}$$

We can still compute the normalization factor in linear time similar to sentence-level log-likelihood.

At inference time, we jointly find the best tag and relation sequence

$$\operatorname*{argmax}_{\widetilde{y}, \widetilde{d}} s(x, \widetilde{y}, \widetilde{d}, \theta')$$

for an input sentence x using Viterbi decoding.

For our task of joint extraction of opinion entities and relations, we train our model to predict tag y and relation distance d for every word in the sentence by maximizing the log-likelihood (**SLL+RLL**) below using Adadelta (Zeiler, 2012).

$$\sum_{(x,y) \in \mathrm{T}} \log p_{sent}(y|x, \theta') + \log p_{rel,Left}(y, d|x, \theta') \tag{3.18}$$

$$+ \log p_{rel,Right}(y, d|x, \theta') \tag{3.19}$$

3.4 Experiments

In this section, we present the datasets used for evaluating the performance of our proposed models. We also present the previous models and compare the performance of our models with them.

	Opinion Expression		
Method	P	R	F1
CRF	**84.42**$^{\mathbf{3.24}}$	61.61$^{3.20}$	*71.17*$^{2.66}$
CRF+ILP	73.53$^{3.90}$	**74.89**$^{\mathbf{2.51}}$	**74.11**$^{\mathbf{2.49}}$
LSTM+WLL	67.88$^{4.49}$	66.13$^{3.20}$	66.87$^{2.66}$
LSTM+SLL	70.45$^{5.12}$	66.65$^{3.46}$	68.37$^{3.14}$
LSTM+SLL+RLL	71.73$^{5.35}$	*70.92*$^{3.96}$	*71.11*$^{2.71}$
CRF	**80.78**$^{\mathbf{3.27}}$	57.62$^{3.24}$	*67.19*$^{2.63}$
CRF+ILP	71.03$^{4.03}$	**69.72**$^{\mathbf{2.37}}$	**70.22**$^{\mathbf{2.44}}$
LSTM+WLL	64.47$^{4.79}$	59.45$^{3.52}$	61.67$^{2.26}$
LSTM+SLL	65.97$^{5.46}$	61.76$^{3.69}$	63.60$^{3.05}$
LSTM+SLL+RLL	65.48$^{4.92}$	*65.54*$^{3.65}$	*65.56*$^{2.71}$

Table 3.1: Performance on opinion expression extraction. Top table shows Binary Overlap performance; bottom table shows Proportional Overlap performance. Superscripts designate one standard deviation.

3.4.1 Data

We use the MPQA 2.0 corpus (Wiebe and Cardie, 2005; Wilson, 2008). It contains news articles and editorials from a wide variety of news sources. There are a total of 482 documents in our dataset containing 9471 sentences with phrase-level annotations. We set aside 132 documents as a development set and use the remaining 350 documents as the evaluation set. We report the results using 10-fold cross validation at the document level to mimic the methodology of Yang and Cardie (2013).

The dataset contains gold-standard annotations for opinion entities — expressions, targets, holders. We use only the direct subjective/opinion expressions. There are also annotations for opinion relations – IS-FROM between opinion holders and opinion expressions; and IS-ABOUT between opinion targets and opinion expressions. These relations can overlap but we discard all relations that contain sub-relations similar to Yang and Cardie (2013). We also leave identification of overlapping relations for future work.

Figure 4.1 gives an example of an annotated sentence from the dataset: boxes denote

	Opinion Target			Opinion Holder		
Method	P	R	F1	P	R	F1
CRF	**80.38**$^{\mathbf{2.72}}$	46.80$^{4.41}$	*59.10*$^{4.06}$	**73.37**$^{\mathbf{4.09}}$	49.71$^{3.46}$	59.21$^{3.49}$
CRF+ILP	*77.27*$^{3.49}$	56.94$^{3.94}$	**65.40**$^{\mathbf{3.07}}$	*67.00*$^{3.17}$	**67.22**$^{\mathbf{3.50}}$	**67.22**$^{\mathbf{2.54}}$
LSTM+WLL	58.71$^{4.87}$	54.92$^{3.23}$	56.50$^{1.51}$	60.33$^{4.54}$	*63.34*$^{2.33}$	61.65$^{2.37}$
LSTM+SLL	63.02$^{4.61}$	56.77$^{3.98}$	*59.65*$^{3.61}$	61.85$^{3.82}$	*63.12*$^{3.59}$	*62.35*$^{2.46}$
LSTM+SLL+RLL	64.52$^{5.52}$	**65.94**$^{\mathbf{4.74}}$	*64.84*$^{1.44}$	62.75$^{3.75}$	*67.17*$^{4.37}$	*64.71*$^{2.23}$
CRF	**71.81**$^{\mathbf{3.22}}$	42.36$^{3.78}$	*53.23*$^{3.69}$	**71.56**$^{\mathbf{3.54}}$	48.61$^{3.51}$	57.86$^{3.43}$
CRF+ILP	**71.94**$^{\mathbf{3.25}}$	49.83$^{3.24}$	**58.72**$^{\mathbf{2.80}}$	*65.70*$^{3.07}$	**65.91**$^{\mathbf{3.63}}$	**65.68**$^{\mathbf{2.61}}$
LSTM+WLL	52.72$^{5.01}$	44.21$^{2.54}$	47.85$^{1.41}$	58.41$^{4.72}$	59.72$^{2.52}$	52.45$^{2.23}$
LSTM+SLL	54.46$^{4.49}$	50.16$^{4.38}$	52.01$^{3.05}$	59.80$^{3.29}$	*61.27*$^{3.75}$	60.40$^{2.26}$
LSTM+SLL+RLL	52.75$^{6.81}$	**60.54**$^{\mathbf{4.78}}$	*55.81*$^{1.96}$	59.44$^{3.56}$	*65.51*$^{4.22}$	*62.18*$^{2.50}$

Table 3.2: Performance on opinion holder and target extraction. Top table shows Binary Overlap performance; bottom table shows Proportional Overlap performance. Superscripts designate one standard deviation.

opinion entities and opinion relations are shown by arcs. We interpret these relations arcs as directed — from an opinion expression towards an opinion holder, and from an opinion target towards an opinion expression.

In order to use the RLL formulation as defined in Section 3.3.3, we pre-process these relation arcs to obtain the left-relation distances (d_{left}) and right-relation distances (d_{right}) as shown in Figure 4.1. For each word in an entity, we find its distance to the nearest word in the related entity. These distances become our relation tags. The entity tags are interpreted using the BIO scheme, also shown in the figure. Our RLL model jointly models the entity tags and relation tags. At inference time, these entity tags and relation tags are used together to determine IS-FROM and IS-ABOUT relations. We use a simple majority vote to determine the final entity tag from SLL+RLL model.

	IS-ABOUT			IS-FROM		
Method	P	R	F1	P	R	F1
CRF+ILP	*61.57*$^{4.56}$	*47.65*$^{3.12}$	*54.39*$^{2.49}$	*64.04*$^{3.08}$	**58.79**$^{\mathbf{4.42}}$	**61.17**$^{\mathbf{3.02}}$
LSTM+SLL+Softmax	36.23$^{5.10}$	36.12$^{7.75}$	35.40$^{3.35}$	36.44$^{5.26}$	40.19$^{6.13}$	37.60$^{3.42}$
LSTM+SLL+RLL	**62.48**$^{\mathbf{3.87}}$	**49.80**$^{\mathbf{2.84}}$	**54.98**$^{\mathbf{2.54}}$	**64.19**$^{\mathbf{3.81}}$	*53.75*$^{6.00}$	*58.22*$^{3.01}$

Table 3.3: Performance on opinion relation extraction using Binary Overlap on the opinion entities. Superscripts designate one standard deviation.

3.4.2 Evaluation Metrics

We use precision, recall and F-measure (as in Yang and Cardie (2013)) as evaluation metrics. Since the identification of exact boundaries for opinion entities is hard even for humans (Wiebe and Cardie, 2005), soft evaluation methods such as Binary Overlap and Proportional Overlap are reported. Binary Overlap counts every overlapping predicted and gold entity as correct, while Proportional Overlap assigns a partial score proportional to the ratio of overlap span and the correct span (Recall) or the ratio of overlap span and the predicted span (Precision).

For the case of opinion relations, we report precision, recall and F-measure according to the Binary Overlap. It considers a relation correct if there is an overlap between the predicted opinion expression and the gold opinion expression as well as an overlap between the predicted entity (holder/target) and the gold entity (holder/target).

3.4.3 Baselines and Previous Models

CRF+ILP. We use the ILP-based joint inference model (Yang and Cardie, 2013) as baseline for both the entity and relation extraction tasks. It represents the state-of-the-art for fine-grained opinion extraction. Their method first identifies opinion entities

using **CRF**s (an additional baseline) with a variety of features such as words, POS tags, and lexicon features (the subjectivity strength of the word in the Subjectivity Lexicon). They also train a relation classifier (logistic regression) by over-generating candidates from the CRFs (50-best paths) using local features such as word, POS tags, subjectivity lexicons as well as semantic and syntactic features such as semantic frames, dependency paths, WordNet hypernyms, etc. Finally, they use ILP for joint-inference to find the optimal prediction for both opinion entity and opinion relation extraction.

LSTM+SLL+Softmax. As an additional baseline for relation extraction, we train a softmax classifier on top of our SLL framework. We jointly learn the relation classifier and SLL model. For every entity pair $[x]_i^j$, $[x]_k^l$, we first sum the start and end word output representation $[z_t]$ and then concatenate them to learn softmax weight W' where $W' \in \mathbb{R}^{3 \times 2d_h}$.

$$y_{rel} = softmax(W^{'} \begin{bmatrix} [z_t]_i + [z_t]_j \\ [z_t]_k + [z_t]_l \end{bmatrix}) \tag{3.20}$$

The inference is pipelined in this case. At the time of inference, we first predict the entity spans and then use these spans for relation classification.

3.4.4 Hyperparameter and Training Details

We use multi-layer bi-directional LSTMs for all the experiments such that the number of hidden layers is 3 and the dimensionality of hidden units (d_h) is 50. We use Adadelta for training. We initialize our word representation using publicly available word2vec (Mikolov et al., 2013) trained on Google News dataset and keep them fixed during training. For RLL, we keep D_{Left} and D_{Right} as 15. All the weights in the network are initialized from small random uniform noise. We train all our models for 200

epochs. We do not pre-train our network. We regularize our network using dropout (Srivastava et al., 2014) with the drop-out rate tuned using the development set. We select the final model based on development-set performance (average of Proportional Overlap for entities and Binary Overlap for relations).

3.5 Results

We discuss the performance of our proposed models in detail with respect to the opinion entities and opinion relation in this section.

3.5.1 Opinion Entities

Table 3.1 shows the performance of opinion entity identification using the Binary Overlap and Proportional Overlap evaluation metrics. We discuss specific results in the paragraphs below.

WLL vs. SLL. SLL performs better than WLL on all entity types, particularly with respect to Proportional Overlap on opinion holder and target entities. A similar trend can be seen for the example sentences in Table 3.5. In S1, SLL extracts "has been in doubt" as the opinion expression whereas WLL only identifies "has". Similarly in S2, WLL annotates "Saudi Arabia's request on a case-by-case" as the target while SLL correctly includes "basis" in its annotation. Thus, we find that modeling the transitions between adjacent tags enables SLL to find entire opinion entity phrases better than WLL, leading to better Proportional Overlap scores.

SLL vs. SLL+RLL. From Table 3.1, we see that the joint-extraction model (SLL+RLL) performs better than SLL as expected. More specifically, SLL+RLL model has better recall for all opinion entity types. The example sentences from Table 3.5 corroborate these results. In S1, SLL+RLL identifies "announced" as an opinion expression, which was missing in both WLL and SLL. In S3, neither the WLL nor the SLL model can annotate opinion holder (H_1) or the target (T_1), but SLL+RLL correctly identifies the opinion entities because of modeling the relations between the opinion expression "will decide" and the holder/target entities.

CRF vs. LSTM-based Models. From the analysis of the performance in Table 3.1, we find that our WLL and SLL models perform worse while our best SLL+RLL model can only match the performance of the CRF baseline on opinion expressions. Even though the recall of all our LSTM-based models is higher than the recall of the CRF-baseline for opinion expressions, we cannot match the precision of CRF baseline. We suspect that the reason for such high precision on the part of the CRF is its access to carefully prepared subjectivity-lexicons[4]. Our LSTM-based models do not rely on such features except via the word-vectors. With respect to holders and targets, we find that our SLL model performs similar to the CRF baseline. However, the SLL+RLL model outperforms CRF baseline.

CRF+ILP vs. SLL+RLL. Even though we find that our LSTM-based joint-model (SLL+RLL) outperforms our LSTM-based only-entity extraction model (SLL), the performance is still below the ILP-based joint-model (CRF+ILP). However, we perform comparably with respect to target entities (Binary Overlap). Also, our recall on targets is much better than all other models whereas the recall on holders is very similar to

[4]http://mpqa.cs.pitt.edu/lexicons/ subj_lexicon/

CRF+ILP. Our SLL+RLL model can identify targets such as "Australia's involvement in Kyoto" which the ILP-based model cannot, as observed for S1 in Table 3.5. In S3, the ILP-based model also erroneously divides the target "consider Saudi Arabia's request on a case-by-case basis" into a holder "Saudi Arabia's" and opinion expression "request", while SLL+RLL model can correctly identify it. We will compare the two models in detail in Section 5.6.

3.5.2 Opinion Relations

The extraction of opinion relations is our primary task. Table 3.3[5] shows the performance on opinion relation extraction task using Binary Overlap.

SLL+Softmax vs. SLL+RLL. The opinion entities and relations are jointly modeled in both the models, but we see a significant improvement in performance by adding relation level dependencies to the model vs. learning a classifier on top of sentence-level dependencies to learn the relation between entities. LSTM+SLL+RLL performs much better in terms of both precision and recall on both IS-FROM and IS-ABOUT relations.

CRF+ILP vs. SLL+RLL. We find that our SLL+RLL model performs comparably and even slightly better on IS-ABOUT relations. Such performance is encouraging because our LSTM-based model does not rely on features such as dependency paths, semantic frames or subjectivity lexicons for our model. Our sequential LSTM model is able to learn these relations thus validating that LSTMs can model long-term dependencies. However, for IS-FROM relations, we find that our recall is lower than the ILP-based

[5] Yang and Cardie (2013) omitted a subset of targets and IS-ABOUT relations. We fixed this and re-ran their models on the updated dataset, obtaining the lower F-score 54.39 for IS-ABOUT relations.

S1 :	However, [Chavez]$_{\mathbf{T_1}}$ who [is known for]$_{\mathbf{O_1}}$ [his]$_{\mathbf{H_2}}$ [ala Fidel Castro left-leaning anti-American philosophy]$_{O_2}$ had on a number of occasions [rebuffed]$_{O_3}$ [[US]$_{\mathbf{H_4}}$ [requests]$_{\mathbf{O_4}}$ for [more oil exports]$_{\mathbf{T_4}}$]$_{\mathbf{T_3}}$.
CRF+ILP	However, [Chavez]$_{\mathbf{H_1}}$ who [is known]$_{\mathbf{O}}$ for [his ala Fidel Castro]$_{\mathbf{H_2}}$ [left-leaning anti-American philosophy]$_{\mathbf{O_2}}$ had on a number of occasions [rebuffed]$_{\mathbf{O_1}}$ [US]$_{\mathbf{H_3}}$ [requests]$_{\mathbf{O_3}}$ for more oil exports.
SLL+RLL	However, Chavez who [is known]$_{\mathbf{O}}$ for his ala Fidel Castro left-leaning anti-American [philosophy]$_{\mathbf{O}}$ had on a number of occasions [rebuffed]$_{\mathbf{O_1}}$ [US requests for more oil exports]$_{\mathbf{T_1}}$.
S2 :	A short while ago, [our correspondent in Bethlehem]$_{\mathbf{H_1}}$ [said]$_{\mathbf{O_1}}$ that [Ra'fat al-Bajjali]$_{\mathbf{T_1}}$ **was martyred** of wounds sustained in the explosion.
CRF+ILP	A short while ago, [our correspondent]$_{\mathbf{H_1}}$ in Bethlehem [said]$_{\mathbf{O_1}}$ that [Ra'fat al-Bajjali]$_{T_1}$ was martyred of wounds sustained in the explosion.
SLL+RLL	A short while ago, our correspondent in Bethlehem said that Ra'fat al-Bajjali was martyred of wounds sustained in the explosion.
S3 :	This is no criticism, and is widely known and appreciated.
CRF+ILP	This is no criticism, and is widely known and appreciated.
SLL+RLL	[This]$_{\mathbf{T_1}}$ [is no criticism]$_{\mathbf{O_1}}$, and is widely [known and appreciated]$_{\mathbf{O}}$.
S4 :	From the fact that mothers care for their young, we can not deduce that they ought to do so, Hume argued.
CRF+ILP	From the fact that [mothers]$_{\mathbf{H_1}}$ [care]$_{\mathbf{O_1}}$ for their young, we can not deduce that they ought to do so, [Hume]$_{\mathbf{H_2}}$ [argued]$_{\mathbf{O_2}}$.
SLL+RLL	From the fact that mothers care for their young, [we]$_{\mathbf{H_1}}$ [can not deduce]$_{\mathbf{O_1}}$ that [they]$_{\mathbf{T_1}}$ ought to do so, [Hume]$_{\mathbf{H_2}}$ [argued]$_{\mathbf{O_2}}$.

Table 3.4: Examples from the dataset with label annotations from CRF+ILP and SLL+RLL models for comparison. The first row for each example is the gold standard.

joint model.

3.6 Discussion

In this section, we discuss the various advantages and disadvantages of the LSTM-based SLL+RLL model as compared to the joint-inference (CRF+ILP) model. We provide examples from the dataset in Table 3.4.

From Table 3.3, we find that SLL+RLL model performs worse with respect to the opinion expression entities and opinion holder entities. On careful analysis of the output, we found cases such as S1 in Table 3.4. For such sentences SLL+RLL model prefers to annotate the opinion target (T_3) "US requests for more oil exports", whereas the ILP model annotates the embedded opinion holder (H_4) "US" and opinion expression (T_4) "requests". Both models are valid with respect to the gold-standard. In order to simplify our problem, we discard these embedded relations during training similar to Yang and Cardie (2013). However, for future work we would like to model these overlapping relations which could potentially improve our performance on opinion holders and opinion expressions.

We also found several cases such as S2, where the SLL+RLL model fails to annotate "said" as an opinion expression. The gold standard opinion expressions include speech events like "said" or "a statement", but not all occurrences of these speech events are opinion expressions, some are merely objective events. In S2, "was martyred" is an indication of an opinion being expressed, so "said" is annotated as an opinion expression. From our observation, the ILP model is more relaxed in annotating most of these speech events as opinion expressions and thus likely to identify corresponding opinion holders and opinion targets as compared to SLL+RLL model.

There were also instances such as S3 and S4 in Table 3.4 for which the gold standard does not have an annotation but the SLL+RLL output looks reasonable with respect to our task. In S3, SLL+RLL identifies "is no criticism" as an opinion expression for the target "This". However, it fails to identify the relation-link between "known and appreciated" and the target "This". Similarly, SLL+RLL also identifies reasonable opinion entities in S4, whereas the ILP model erroneously annotates "mothers" as the opinion holder and "care" as the opinion expression.

We handle the task of joint-extraction of opinion entities and opinion relations as a sequence labeling task in the work described in the work described in this chapter and report the performance of the 1-best path at the time of Viterbi inference. However, there are approaches such as discriminative reranking (Collins and Koo, 2005) to rerank the output of an existing system that offer a means for further improving the performance of our SLL+RLL model. In particular, the oracle performance using the top-10 Viterbi paths from our SLL+RLL model has an F-score of 82.11 for opinion expressions, 76.77 for targets and 78.10 for holders. Similarly, IS-ABOUT relations have an F-score of 65.99 and IS-FROM relations, an F-score of 70.80. These scores are on average 10 points better than the performance of the current SLL+RLL model, indicating that substantial gains might be attained via reranking.

3.7 Chapter Summary

In the work described in this chapter, we explored LSTM-based models for the joint extraction of opinion entities and relations. Experimentally, we found that adding sentence-level and relation-level dependencies on the output layer improves the performance on opinion entity extraction, obtaining results within 1-3% of the ILP-based joint model on opinion entities, within 3% for IS-FROM relation and comparable for IS-ABOUT relation.

Going forward, it would be interesting to explore the effects of pre-training (Bengio et al., 2009) and scheduled sampling (Bengio et al., 2015) for training our LSTM network. Our approach can also be potentially benefited by re-ranking methods. With respect to the fine-grained opinion mining task, a potential direction to improve the models to be able to model overlapping and embedded entities and relations. We will discuss

our approaches for identifying overlapping and embedded entities in Chapter 5 which can benefit further research in the problem of opinion mining. Also, these models can be benefited from handling cross-sentential relations. In our work (Bommasani et al., 2019; Niculae et al., 2016; Chen et al., 2014) we also present approaches for identifying opinion relations across sentences. We also present models (Park et al., 2015) for identifying appropriate types of support for propositions comprising online user comments.

S1 :	[Australia's involvement in Kyoto]$_{\mathbf{T_1}}$ [has been in doubt]$_{\mathbf{O_1}}$ ever since [the US President, George Bush]$_{\mathbf{H_2}}$, [announced]$_{\mathbf{O_2}}$ last year that [ratifying the protocol]$_{\mathbf{T_2}}$ would hurt the US economy.
CRF+ILP	Australia's involvement in Kyoto [has been in doubt]$_{\mathbf{O_1}}$ ever since the US President, George Bush, announced last year that [ratifying the protocol]$_{\mathbf{T_1}}$ would hurt ...
WLL	[Australia's involvement in Kyoto]$_{\mathbf{T}}$ [has]$_{\mathbf{O}}$ been in doubt ever since the US [President]$_{\mathbf{H}}$, [George Bush]$_{\mathbf{H}}$, announced last year that ratifying the protocol ....
SLL	[Australia's involvement in Kyoto]$_{\mathbf{T}}$ [has been in doubt]$_{\mathbf{O}}$ ever since the US President, George Bush, announced last year that ratifying the protocol would hurt ...
SLL+RLL	[Australia's involvement in Kyoto]$_{\mathbf{T}}$ [has been in doubt]$_{\mathbf{O}}$ ever since the US President, [George Bush]$_{\mathbf{H_2}}$, [announced]$_{\mathbf{O_2}}$ last year that [ratifying the protocol]$_{\mathbf{T_2}}$ ...
S2 :	Bush said last week [he]$_{\mathbf{H_{1,2}}}$ [was willing]$_{\mathbf{O_1}}$ [to consider]$_{\mathbf{O_2}}$ [Saudi Arabia's request on a case-by-case basis]$_{\mathbf{T_2}}$ but [U.S. officials]$_{\mathbf{H_3}}$ [doubted]$_{\mathbf{O_3}}$ [it would happen any time soon]$_{\mathbf{T_3}}$.
CRF+ILP	[Bush]$_{\mathbf{H_1}}$ [said]$_{\mathbf{O_1}}$ last week [he]$_{\mathbf{H_2}}$ [was willing to consider]$_{\mathbf{O_2}}$ [Saudi Arabia's]$_{\mathbf{H_3}}$ [request]$_{\mathbf{O_3}}$ on a case-by-case basis but [U.S. officials]$_{\mathbf{H_4}}$ [doubted]$_{\mathbf{O_4}}$ [it]$_{\mathbf{T_4}}$ would happen any time soon.
WLL	Bush said last week [he]$_{\mathbf{H}}$ [was willing]$_{\mathbf{O}}$ to [consider]$_{\mathbf{O}}$ [Saudi Arabia's request on a case-by-case]$_{\mathbf{T}}$ basis but [U.S. officials]$_{\mathbf{H}}$ [doubted]$_{\mathbf{O}}$ [it]$_{\mathbf{T}}$ would [happen any time soon]$_{\mathbf{T}}$.
SLL	Bush said last week [he]$_{\mathbf{H}}$ [was willing]$_{\mathbf{O}}$ to [consider Saudi Arabia's request on a case-by-case basis]$_{\mathbf{T}}$ but [U.S. officials]$_{\mathbf{H}}$ [doubted]$_{\mathbf{O}}$ [it]$_{\mathbf{T}}$ would happen any time soon.
SLL+RLL	Bush said last week [he]$_{\mathbf{H_1}}$ [was willing to consider]$_{\mathbf{O_1}}$ [Saudi Arabia's request on a case-by-case basis]$_{\mathbf{T_1}}$ but [U.S. officials]$_{\mathbf{H_2}}$ [doubted]$_{\mathbf{O_2}}$ [it would happen any time soon]$_{\mathbf{T_2}}$.
S3 :	Hence, [the Organization of Petroleum Exporting Countries (OPEC)]$_{\mathbf{H_1}}$, [will decide]$_{\mathbf{O_1}}$ at its meeting on Wednesday [whether or not to cut its worldwide crude production in an effort to shore] [up energy prices]$_{\mathbf{T_1}}$.
CRF+ILP	Hence, the Organization of Petroleum Exporting Countries (OPEC), [will decide]$_{\mathbf{O_1}}$ at its meeting on Wednesday whether [or not to cut its worldwide crude production in an effort to shore up energy prices]$_{\mathbf{T_1}}$.
WLL	Hence, the Organization of Petroleum Exporting Countries (OPEC), will [decide]$_{\mathbf{O}}$ at its meeting on Wednesday whether or not to cut its worldwide crude production in an effort to shore up energy prices.
SLL	Hence, the Organization of Petroleum Exporting Countries (OPEC), [will decide]$_{\mathbf{O}}$ at its meeting on Wednesday whether or not to cut its worldwide crude production in an effort to shore up energy prices.
SLL+RLL	Hence, [the Organization of Petroleum Exporting Countries (OPEC)]$_{\mathbf{H_1}}$, [will decide]$_{\mathbf{O_1}}$ at its meeting on Wednesday whether [or not to cut its worldwide crude production in an effort to shore up energy prices]$_{\mathbf{T_1}}$.

Table 3.5: Output from different models. The first row for each example is the gold standard.

CHAPTER 4

JOINT EXTRACTION OF ENTITIES AND RELATIONS

Contrary to the previous chapter where we proposed methods for extracting opinions from text, in this chapter we propose a model for extracting facts from sentences in a more general information extraction setting for extracting facts. We show that the work described in this chapter overcomes some of the limitations of the model described in Chapter 3. This chapter is based on the work described in Katiyar and Cardie (2017). We will first provide an overview of the recent approaches for this problem and then describe our proposed approach.

As we briefly discussed in Chapter 2, extraction of entities and their relations from text belongs to a very well-studied family of structured prediction tasks in NLP. There are several NLP tasks such as fine-grained opinion mining (Choi et al., 2006) also discussed in Chapter 3, semantic role labeling (Gildea and Jurafsky, 2002), etc., which have a similar structure; thus making it an important and a challenging task.

Several methods have been proposed for entity mention and relation extraction at the sentence-level. These can be broadly categorized into – 1) pipeline models that treat the identification of entity mentions (Nadeau and Sekine, 2007) and relation classification (Zhou et al., 2005) as two separate tasks; and 2) joint models, also the more recent, which simultaneously identify the entity mention and relations (Li and Ji, 2014; Miwa and Sasaki, 2014). Joint models have been argued to perform better than the pipeline models as knowledge of the typed relation can increase the confidence of the model on entity extraction and vice versa.

Recurrent networks (RNNs) (Elman, 1990) have recently become very popular for sequence tagging tasks such as entity extraction that involves a set of contiguous tokens.

However, their ability to identify relations between *non-adjacent* tokens in a sequence, e.g., the head nouns of two entities, is less explored. For these tasks, RNNs that make use of tree structures have been deemed more suitable. Miwa and Bansal (2016a), for example, propose an RNN comprised of a sequence-based long short term memory (LSTM) for entity identification and a separate tree-based dependency LSTM layer for relation classification using shared parameters between the two components. As a result, their model depends critically on access to dependency trees, restricting it to sentence-level extraction and to languages for which (good) dependency parsers exist. Also, their model does not jointly extract entities and relations; they first extract all entities and then perform relation classification on all pairs of entities in a sentence.

In our previous work (Katiyar and Cardie, 2016) described in Chapter 3, we address the same task in an opinion extraction context. Our LSTM-based formulation explicitly encodes distance between the head of entities into opinion relation labels. The output space of our model is quadratic in size of the entity and relation label set and we do not specifically identify the relation type. Unfortunately, adding relation type makes the output label space very sparse, making it difficult for the model to learn.

In the work described in this chapter, we propose a novel RNN-based model for the joint extraction of entity mentions and relations. Unlike other models, our model does not depend on any dependency tree information. Our RNN-based model is a multi-layer bi-directional LSTM over a sequence. We encode the output sequence from left-to-right. At each time step, we use an attention-like model on the previously decoded time steps, to identify the tokens in a specified relation with the current token. We also add an additional layer to our network to encode the output sequence from right-to-left and find significant improvement on the performance of relation identification using bi-directional encoding.

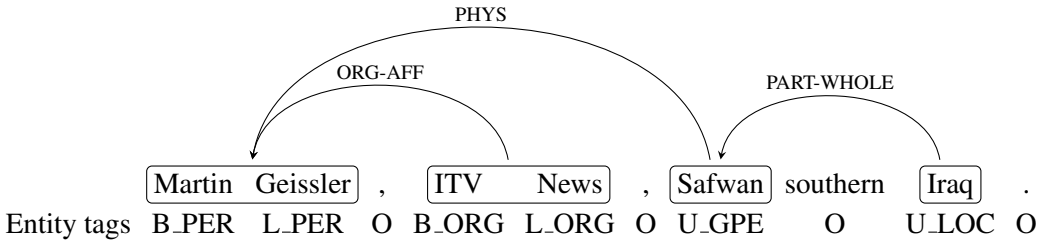


Figure 4.1: Gold standard annotation for an example sentence from ACE05 dataset.

Our model significantly outperforms the feature-based structured perceptron model of Li and Ji (2014), showing improvements on both entity and relation extraction on the ACE05 dataset. In comparison to the dependency tree-based LSTM model of Miwa and Bansal (2016a), our model performs within 1% on entities and 2% on relations on ACE05 dataset. We also find that our model performs significantly better than their tree-based model on the AGENT-ARTIFACT relation, while their tree-based model performs better on PHYSICAL and PART-WHOLE relations; the two models perform comparably on all other relation types. The very competitive performance of our non-tree-based model bodes well for relation extraction of non-adjacent entities in low-resource languages that lack good parsers.

In the sections that follow, we describe related work (Section 4.1); our bi-directional LSTM model with attention (Section 6.2); the training (Section 6.3); the experiments on ACE dataset (Section 6.4); results (Section 6.5); error analysis (Section 5.6) and conclusion (Section 5.7).

4.1 Related Work

RNNs (Hochreiter and Schmidhuber, 1997) have been recently applied to many sequential modeling and prediction tasks, such as machine translation (Bahdanau et al., 2015;

Sutskever et al., 2014), named entity recognition (NER) (Hammerton, 2003), opinion mining (İrsoy and Cardie, 2014). Variants such as adding CRF-like objective on top of LSTMs have been found to produce state-of-the-art results on several sequence prediction NLP tasks (Collobert et al., 2011; Huang et al., 2015; Katiyar and Cardie, 2016) as seen in Chapter 3. These models assume conditional independence at the output layer whereas the model we propose in the work described in this chapter does not assume any conditional independence at the output layer, allowing it to model an arbitrary distribution over output sequences.

Relation classification has been widely studied as a stand-alone task, assuming that the arguments of the relations are known in advance. There have been several models proposed including feature-based models (Bunescu and Mooney, 2005; Zelenko et al., 2003) and neural network based models (Socher et al., 2012; dos Santos et al., 2015; Hashimoto et al., 2015; Xu et al., 2015a,b).

For joint-extraction of entities and relations, feature-based structured prediction models (Li and Ji, 2014; Miwa and Sasaki, 2014), joint inference integer linear programming models(Yih and Roth, 2007; Yang and Cardie, 2013), card-pyramid parsing (Kate and Mooney, 2010) and probabilistic graphical models (Yu and Lam, 2010; Singh et al., 2013) have been proposed. In contrast, we propose a neural network model which does not depend on the availability of any features such as part of speech (POS) tags, dependency trees, etc.

Recently, Miwa and Bansal (2016a) proposed an end-to-end LSTM based sequence and tree-structured model. They extract entities via a sequence layer and relations between the entities via the shortest path dependency tree network. In this chapter, we try to investigate recurrent neural networks with attention for extracting semantic relations between entity mentions without using any dependency parse tree features. We also

present the *first* neural network based joint model that can extract entity mentions and relations along with the relation type. In our previous work (Katiyar and Cardie, 2016), as explained earlier in Chapter 3, we proposed a LSTM-based model for joint extraction of opinion entities and relations, but no relation types. This model cannot be directly extended to include relation types as the output space becomes sparse making it difficult for the model to learn.

Recent advances in recurrent neural network has seen the application of attention on recurrent neural networks to obtain a representation weighted by the importance of tokens in the sequence model. Such models have been very frequently used in question-answering tasks (for recent examples, see Chen et al. (2016) and Lee et al. (2016)), machine translation (Luong et al., 2015; Bahdanau et al., 2015), and many other NLP applications. Pointer networks (Vinyals et al., 2015a), an adaptation of attention models, use these token-level weights as pointers to the input elements. Zhai et al. (2017), for example, have used these for neural chunking, and Nallapati et al. (2016) and Cheng and Lapata (2016), for summarization. However, to the best of our knowledge, these networks have not been used for joint extraction of entity mentions and relations. We present first such attempt to use these attention models with recurrent neural networks for joint extraction of entity mentions and relations.

4.2 Model

Our model comprises of a multi-layer bi-directional recurrent network which learns a representation for each token in the sequence. We use the hidden representation from the top layer for joint entity and relation extraction. For each token in the sequence, we output an entity tag and a relation tag. The entity tag corresponds to the entity type,

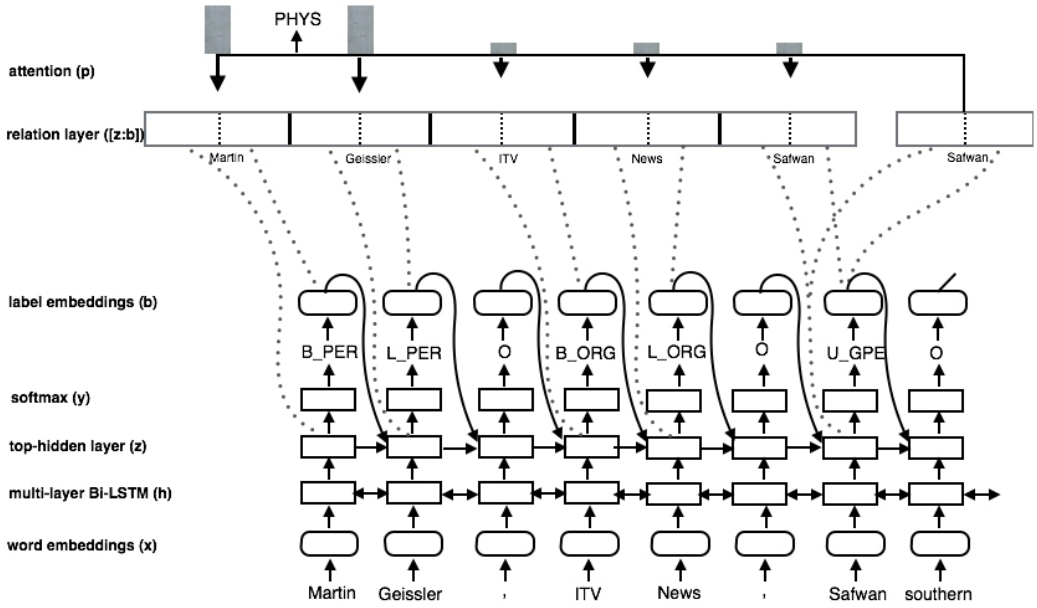


Figure 4.2: Our network structure based on bi-directional LSTMs for joint entity and relation extraction. This snapshot shows the network when encoding the relation tag for the word "Safwan" in the sentence. The dotted lines in the figure show that top hidden layer and label embeddings for tokens is copied into relation layer. The pointers at attention layer indicate the probability distribution over tokens, the length of the pointers is used to denote the probability value.

whereas the relation tag is a tuple of pointers to related entities and their respective relation types. Figure 4.1 shows the annotation for an example sentence from the dataset. We transform the relation tags from entity level to token level. For example, we separately model the relation "ORG-AFF" for each token in the entity "ITV News". Thus, we model the relations between "ITV" and "Martin Geissler", and "News" and "Martin Geissler" separately. We employ a pointer-like network on top of the sequence layer in order to find the relation tag for each token as shown in Figure 4.2. At each time step, the network utilizes the information available about all output tags from the previous time steps in order to output the entity tag and relation tag *jointly* for the current token.

4.2.1 Multi-layer Bi-directional Recurrent Network

We use multi-layer bi-directional LSTMs for sequence tagging because LSTMs are more capable of capturing long-term dependencies between tokens, making it ideal for both entity mention and relation extraction.

Using LSTMs, we can compute the hidden state $\overrightarrow{h_t}$ in the forward direction and $\overleftarrow{h_t}$ in the backward direction for every token as below:

$$\overrightarrow{h}_t = LSTM(x_t, \overrightarrow{h}_{t-1}) \tag{4.1}$$

$$\overleftarrow{h}_t = LSTM(x_t, \overleftarrow{h}_{t+1}) \tag{4.2}$$

For every token t in the subsequent layer l, we combine the representations $\overrightarrow{h}_t^{l-1}$ and $\overleftarrow{h}_t^{l-1}$ from previous layer l-1 and feed it as an input. In this work, we only use the hidden state from the last layer L for output layer and compute the top hidden layer representation as below:

$$z_t^{'} = \overrightarrow{V}\overrightarrow{h}_t^{(L)} + \overleftarrow{V}\overleftarrow{h}_t^{(L)} + c \tag{4.3}$$

$\overrightarrow{V}$ and $\overleftarrow{V}$ are weight matrices for combining hidden representations from the two directions.

4.2.2 Entity detection

We formulate entity detection as a sequence labeling task using BILOU scheme similar to Li and Ji (2014) and Miwa and Bansal (2016a). We assign each token in the entity with the tag B appended with the entity type if it is the beginning of the entity, I for inside of an entity, L for the end of the entity or U if there is only one token in the entity. Figure 4.1 shows an example of the entity tag sequence assigned to the sentence. For

each token in the sequence, we perform a softmax over all candidate tags to output the most likely tag:

$$y_t = \text{softmax}(Uz_t^{'} + b) \tag{4.4}$$

Our network structure as shown in Figure 4.2 also contains connections from the output y_{t-1} of the previous time step to the current top hidden layer. Thus our outputs are not conditionally independent from each other. In order to add connections from y_{t-1}, we transform this output k into a label embedding b_{t-1}^k [1]. We represent each label type k with a dense representation b^k. We compute the output layer representations as:

$$z_t = LSTM([z_t^{'}; b_{t-1}^k], h_{t-1}) \tag{4.5}$$

$$y_t = \text{softmax}(Uz_t + b^{'}) \tag{4.6}$$

We decode the output sequence from left to right in a greedy manner.

4.2.3 Attention Model

We use attention model for relation extraction. Attention models, over an encoder sequence of representations z, can compute a soft probability distribution p over these learned representations, where d_i is the ith token in decoder sequence. These probabilities are an indication of the importance of different tokens in the encoder sequence:

$$u_t^i = v^T \tanh(W_1 z + W_2 d_i) \tag{4.7}$$

$$p_t^i = \text{softmax}(u_t^i) \tag{4.8}$$

v is a weight matrix for attention which transforms the hidden representations into attention scores.

[1] We can also add relation label embeddings using the relation tag output from the previous time step.

We use pointer networks (Vinyals et al., 2015a) in our approach, which are a variation of these attention models. Pointer networks interpret these p_t^i as the probability distribution over the input encoding sequence and use u_t^i as pointers to the input elements. We can use these pointers to encode relation between the current token and the previous predicted tokens, making it fit for relation extraction as explained in Section 4.2.4.

4.2.4 Relation detection

We formulate relation extraction also as a sequence labeling task. For each token, we want to find the tokens in the past that the current token is related to along with its relation type. In Figure 4.1, "Safwan" is related to the tokens "Martin" as well as "Geissler" by the relation type "PHYS". For simplicity, let us assume that there is only one previous token the current token is related to when training, i.e., "Safwan" is related to "Geissler" via PHYS relation. We can extend our approach to output multiple relations as explained in Section 6.3.

We use pointer networks as described in Section 4.2.3. At each time step, we stack the top hidden layer representations from the previous time steps $z_{\leq t}$[2] and its corresponding label embeddings $b_{\leq t}$. We only stack the top hidden layer representations for the tokens which were predicted as non-O's for previous time steps as shown in Figure 4.2. Our decoding representation at time t is the concatenation of z_t and b_t. The

[2]The notation $\leq$ is used to denote the stacking of the representations from the previous time steps. Thus, if z_t is a 2-dimensional matrix then $z_{\leq t}$ will be a 3-dimensional tensor. The size along the first dimension will now correspond to the number of 2-dimensional matrices stacked.

attention probabilities can now be computed as below:

$$u^t_{\leq t} = v^T \tanh(W_1[z_{\leq t}; b_{\leq t}] + W_2[z_t; b_t]) \tag{4.9}$$

$$p^t_{\leq t} = \text{softmax}(u^t_{\leq t}) \tag{4.10}$$

Thus, $p^t_{\leq t}$ corresponds to the probability of each token, in the sequence so far, being related to the current token at time step t. For the case of NONE relations, the token at t is related to itself.

We also want to find the type of the relations. In order to achieve this, we add an extra dimension to v corresponding to the size of relation types R space. Thus, u^i_t is no longer a score but a R dimensional vector. We then take softmax over this vector of size $O(|z_{\leq t}| \times R)$ to find the most likely tuple of pointer to the related entity and its relation type.

4.2.5 Bi-directional Encoding

Bi-directional LSTMs have been found to be able to capture context better than plain left-to-right LSTMs, based on their performance on various NLP tasks (İrsoy and Cardie, 2014). Also, Sutskever et al. (2014) found that their performance on machine translation task improved on reversing the input sentences during training. Inspired by these developments, we experiment with bi-directional encoding at the output layer. We add another top hidden layer on Bi-LSTM in Figure 4.2 which encodes the output sequence from right-to-left. The two encoding share the same multi-layer bi-directional LSTM except for the top hidden layer. Thus, we have two output layers in our network which output the entity tags and relation tags separately. At inference time, we employ heuristics to combine the output from the two directions.

4.3 Training

We train our network by maximizing the log-probability of the correct entity E and relation R tag sequences *jointly* given the sentence S as below:

$$\log p(E, R|S, \theta) \tag{4.11}$$

$$= \frac{1}{|S|} \sum_{i \in |S|} \log p(e_i, r_i | e_{<i}, r_{<i}, S, \theta) \tag{4.12}$$

$$= \frac{1}{|S|} \sum_{i \in |S|} \log p(e_i | e_{<i}, r_{<i}) + \log p(r_i | e_{\leq i}, r_{<i}) \tag{4.13}$$

Thus, we can decompose our objective into the sum of log-probabilities over entity sequence and relation sequence. We use the gold entity tags while training. As shown in Figure 4.2, we input the label embedding from the previous time step to the top hidden layer at the current time step along with the other recurrent inputs. During training, we pass the gold label embedding to the next time step which enables better training of our model. However, at test time when the gold label is not available we use the predicted label at previous time step as input to the current step.

At inference time, we can greedily decode the sequence to find the most likely entity $\widehat{E}$ and relation $\widehat{R}$ tag sequences:

$$(\widehat{E}, \widehat{R}) = \underset{E,R}{\operatorname{argmax}}\ p(E, R) \tag{4.14}$$

Since, we add another top layer to encode tag sequences in the reverse order as explained in Section 4.2.5, there may be conflicts in the output. We select the positive and more confident label similar to Miwa and Bansal (2016a).

Multiple Relations Our approach to relation extraction is different from Miwa and Bansal (2016a). Miwa and Bansal (2016a) present each pair of entities to their model for relation classification. In our approach, we use pointer networks to identify the

related entities. Thus, for our approach described so far if we only compute the argmax on our objective then we limit our model to output only one relation label per token. However, from our analysis of the dataset, an entity may be related to more than one entity in the sentence. Hence, we modify our objective to include multiple relations. In Figure 4.2, token "Safwan" is related to both tokens "Martin" and "Geissler" of the entity "Martin Geissler", hence we assign probability of 0.5 to both these tokens. This can be easily expanded to include tokens from other related entities, such that we assign equal probability $\frac{1}{N}$ to all tokens[3] depending on the number N of these related tokens.

The log-probability for the entity part remain the same as in our objective discussed in Section 6.3, however we modify the relation log-probability as below:

$$\sum_{|j:\mathbf{r}'_{\mathbf{i,j}}>0|} \mathbf{r}'_{\mathbf{i,j}} \log p(\mathbf{r_{i,j}}|e_{\leq i}, \mathbf{r_{<i}}, S, \theta) \tag{4.15}$$

where, $\mathbf{r}'_{\mathbf{i}}$ is the true distribution over relation label space and $\mathbf{r_i}$ is the softmax output from our model. From empirical analysis, we find that $\mathbf{r}'_{\mathbf{i}}$ is generally sparse and hence using a cross entropy objective like this can be useful to find multiple relations. We can also use Sparsemax (Martins and Astudillo, 2016) instead of softmax which is more suitable for sparse distributions. However, we leave it for future work.

At inference time, we output all the labels with probability value above a certain threshold. We adapt this threshold based on the validation set.

4.4 Experiments

In this section, we present the datasets used for evaluating the performance of our proposed model. We also present the previous models and compare the performance of our

[3]In this work, we only identify mention heads and hence the span is limited to a few tokens. We can also include only the last token of the gold entity span in the gold probability distribution.

	Entity			Relation			Entity+Relation		
Method	P	R	F1	P	R	F1	P	R	F1
Li and Ji (2014)	.852	.769	.808	.689	.419	.521	.654	.398	.495
SPTree	.829	.839	.834	–	–	–	.572	.540	.556
SPTree[1]	.823	.839	.831	.605	.553	.578	.578	.529	.553
Our Model	.840	.813	.826	.579	.540	.559	.555	.518	.536

Table 4.1: Performance on ACE05 test dataset. The dashed ("–") performance numbers were missing in the original paper (Miwa and Bansal, 2016a).

[1] We ran the system made publicly available by Miwa and Bansal (2016a) on ACE05 dataset for filling in the missing values and comparing our system with theirs at fine-grained level.

models with them.

4.4.1 Data

We evaluate our proposed model on the two datasets from the Automatic Content Extraction (ACE) program – ACE05 and ACE04. There are 7 main entity types namely Person (PER), Organization (ORG), Geographical Entities (GPE), Location (LOC), Facility (FAC), Weapon (WEA) and Vehicle (VEH). For each entity, both entity mentions and its head phrase are annotated. For the scope of this work, we only use the entity head phrase similar to Li and Ji (2014) and Miwa and Bansal (2016a). Also, there are relation types namely Physical (PHYS), Person-Social (PER-SOC), Organization-Affiliation (ORG-AFF), Agent-Artifact (ART), GPE-Affiliation (GPE-AFF).

ACE05 has a total of 6 relation types including PART-WHOLE. We use the same data splits as Li and Ji (2014) and Miwa and Bansal (2016a) such that there are 351 documents for training, 80 for development and the remaining 80 documents for the test

	Entity			Relation			Entity+Relation		
Method	P	R	F1	P	R	F1	P	R	F1
Li and Ji (2014)	.835	.762	.797	.647	.385	.483	.608	.361	.453
SPTree	.808	.829	.818	–	–	–	.487	.481	.484
Our Model	.812	.781	.796	.502	.488	.493	.464	.453	.457

Table 4.2: Performance on ACE04 test dataset. The dashed ("–") performance numbers were missing in the original paper (Miwa and Bansal, 2016a).

set.

ACE04 has 7 relation types with an additional Discourse (DISC) type and split ORG-AFF relation type into ORG-AFF and OTHER-AFF. We perform 5-fold cross validation similar to Chan and Roth (2011) for fair comparison with the state-of-the-art.

4.4.2 Evaluation Metrics

In order to compare our system with the previous systems, we report micro F1-scores, Precision and Recall on both entities and relations similar to Li and Ji (2014) and Miwa and Bansal (2016a). An entity is considered correct if we can identify its head and the entity type correctly. A relation is considered correct if we can identify the head of the argument entities and also the relation type. We also report a combined score when both argument entities and relations are correct.

4.4.3 Baselines and Previous Models

We compare our approach with two previous approaches. The model proposed by Li and Ji (2014) is a feature-based structured perceptron model with efficient beam-search.

	Entity			Relation			Entity+Relation		
Encoding	P	R	F1	P	R	F1	P	R	F1
Left-to-Right	.821	.812	.817	.622	.449	.522	.601	.434	.504
+Multiple Relations	.835	.811	.823	.560	.492	.524	.539	.473	.504
+Bi-directional (Our Model)	.840	.813	.826	.579	.540	.559	.555	.518	.536

Table 4.3: Performance of different encoding methods on ACE05 dataset.

They employ a segment-based decoder instead of token-based decoding. Their model outperformed previous state-of-the-art pipelined models. Miwa and Sasaki (2014) (SP-Tree) recently proposed a LSTM-based model with a sequence layer for entity identification, and a tree-based dependency layer which identifies relations between pairs of candidate entities using the shortest dependency path between them. We also employed our previous approach (Katiyar and Cardie, 2016) for extraction of opinion entities and relations to this task. We found that the performance was not competitive with the two approaches mentioned above, performing upto 10 points lower on relations. Hence, we do not include the results in Table 4.1. Also, Li and Ji (2014) showed that the joint model performs better than the pipelined approaches. Thus, we do not include any pipeline baselines.

4.4.4 Hyperparameters and Training Details

We train our model using Adadelta (Zeiler, 2012) with gradient clipping. We regularize our network using dropout (Srivastava et al., 2014) with the drop-out rate tuned using development set. We initialized our word embeddings with 300-dimensional word2vec (Mikolov et al., 2013) word embeddings trained on Google News dataset. We have 3 hidden layers in our network and the dimensionality of the hidden units is 100. All the weights in the network are initialized from small random uniform noise. We tune our

hyperparameters based on ACE05 development set and use them for training on ACE04 dataset.

4.5 Results

Table 4.1 compares the performance of our system with respect to the baselines on ACE05 dataset. We find that our joint model significantly outperforms the joint structured perceptron model (Li and Ji, 2014) on both entities and relations, despite the unavailability of features such as dependency trees, POS tags, etc. However, if we compare our model to the SPTree models, then we find that their model has better recall on both entities and relations. In Section 5.6, we perform error analysis to understand the difference in the performance of the two models in detail.

We also compare the performance of various encoding schemes in Table 4.3. We compare the benefits of introducing multiple relations in our objective and bi-directional encoding compared to left-to-right encoding.

Multiple Relations We find that modifying our objective to include multiple relations improves the recall of our system on relations, leading to slight improvement on the overall performance on relations. However, careful tuning of the threshold may further improve precision.

Bi-directional Encoding By adding bi-directional encoding to our system, we find that we can significantly improve the performance of our system compared to left-to-right encoding. It also improves precision compared to left-to-right decoding combined with multiple relations objective.

We find that for some relations it is easier to detect them with respect to one of the entities in the entity pair. PHYS relation is easier identified with respect to GPE entity than PER entity. Thus, our bi-directional encoding of relations allows us to encode these relations with respect to both entities in the relation.

Table 4.2 shows the performance of our model on ACE04 dataset. We believe that tuning the hyperparameters of our model can further improve the results on this dataset. As also pointed out by Li and Ji (2014) that ACE05 has better annotation quality, we focused on ACE05 dataset for this work.

4.6 Error Analysis

In this section, we perform a fine-grained comparison of our model with respect to the SPTree (Miwa and Bansal, 2016a) model. We compare the performance of the two models with respect to entities, relation types and the distance between the relation arguments and provide examples from the test set in Table 4.5.

4.6.1 Entities

We find that our model has lower recall on entity extraction than SPTree as shown in Table 4.1. Miwa and Bansal (2016a), in one of the ablation tests on ACE05 development set, show that their model can gain upto 2% improvement in recall by entity pretraining. Since we propose a joint-model, we cannot directly apply their pretraining trick on entities separately. We leave it for future work. Li and Ji (2014) mentioned in their analysis of the dataset that there were many "UNK" tokens in the test set which were never seen during training. We verified the same and we hypothesize that for this reason the

performance on the entities depends largely on the pretrained word embeddings being used. We found considerable improvements on entity recall when using pretrained word embeddings, if available, for these "UNK" tokens. Miwa and Bansal (2016a) also use additional features such as POS tags in addition to pretrained word embeddings at the input layer.

Relation Type	Method	R	P	F1
ART	SPTree	.363	.552	.438
	Our model	.431	.611	**.505**
PART-WHOLE	SPTree	.560	.538	**.548**
	Our model	.520	.538	.528
PER-SOC	SPTree	.671	.671	.671
	Our model	.657	.648	.652
PHYS	SPTree	.489	.513	**.500**
	Our model	.388	.426	.406
GEN-AFF	SPTree	.414	.640	.502
	Our model	.484	.516	.500
ORG-AFF	SPTree	.692	.704	.697
	Our model	.706	.700	.703

Table 4.4: Performance on different relation types in ACE05 test dataset.

4.6.2 Relation Types

We evaluate our model on different relation types and compare the performance with SPTree model in Table 4.4. Interestingly, we find that the performance of the two models is varied over different relation types. The dependency tree-based model significantly outperforms our joint-model on PHYS and PART-WHOLE relations, whereas our model is significantly better than tree-based model on ART relation. We show an example sentence (S1) in Table 4.5, where SPTree model identifies the entities in ART relation correctly but fails to identify ART relation. We compare the performance with respect to PHYS relation in Section 4.6.3.

S1 :	the [men]$_{\text{PER:ART-1}}$ held on the sinking [vessel]$_{\text{VEH:ART-1}}$ until the [passenger]$_{\text{PER:ART-2}}$ [ship]$_{\text{VEH:ART-2}}$ was able...
SPTree :	the [men]$_{\text{PER}}$ held on the sinking [vessel]$_{\text{VEH}}$ until the [passenger]$_{\text{PER}}$ [ship]$_{\text{VEH}}$ was able to reach them.
Our Model :	the [men]$_{\text{PER:ART-1}}$ held on the sinking [vessel]$_{\text{VEH:ART-1}}$ until the [passenger]$_{\text{PER:ART-2}}$ [ship]$_{\text{VEH:ART-2}}$ was able...
S2 :	[her]$_{\text{PER}}$ research was conducted [here]$_{\text{FAC}}$ at a [location]$_{\text{FAC:PHYS1}}$ well-known to [u.n.]$_{\text{ORG:ORG-AFF1}}$ [arms]$_{\text{WEA}}$ [inspectors]$_{\text{PER:ORG-AFF1}}$. 300 miles west of [baghdad]$_{\text{GPE:PHYS1}}$.
SPTree :	[her]$_{\text{PER}}$ research was conducted [here]$_{\text{GPE}}$ at a [location]$_{\text{LOC:PHYS1}}$ well-known to u.n. [arms]$_{\text{WEA}}$ [[inspectors]$_{\text{PER:PHYS1,PHY2}}$. 300 miles west of [baghdad]$_{\text{GPE:PHYS2}}$.
Our Model :	[her]$_{\text{PER}}$ research was conducted [here]$_{\text{FAC:PHYS1}}$ at a [location]$_{\text{GPE}}$ well-known to [u.n.]$_{\text{ORG:ORG-AFF1}}$ [arms]$_{\text{WEA}}$ [inspectors]$_{\text{PER:ORG-AFF1}}$. 300 miles west of [baghdad]$_{\text{GPE:PHYS1}}$.
S3 :	... [Abigail Fletcher]$_{\text{PER:PHYS1}}$, a [marcher]$_{\text{FAC:GEN-AFF2}}$ from [Florida]$_{\text{FAC:GEN-AFF2}}$, said outside the [president]$_{\text{PER:ART3}}$'s [[residence]$_{\text{FAC:ART3, PHYS1}}$.
SPTree :	... [Abigail Fletcher]$_{\text{PER:PHYS1}}$, a [marcher]$_{\text{FAC:GEN-AFF2}}$ from [Florida]$_{\text{FAC:GEN-AFF2}}$, said outside the [president]$_{\text{PER:ART3}}$'s [[residence]]$_{\text{FAC:ART3, PHYS1}}$.
Our Model :	... [Abigail Fletcher]$_{\text{PER}}$, a [marcher]$_{\text{FAC:GEN-AFF2}}$ from [Florida]$_{\text{FAC:GEN-AFF2}}$, said outside the [president]$_{\text{PER}}$'s residence.

Table 4.5: Examples from the dataset with label annotations from SPTree and our model for comparison. The first row for each example is the gold standard.

		Relation		
Distance	Method	R	P	F1
≤ 7	SPTree	.589	.628	.608
	Our model	.591	.605	.598
> 7	SPTree	.275	.375	**.267**
	Our model	.153	.259	.192

Table 4.6: Performance based on the distance between entity arguments in relations for ACE05 test dataset.

4.6.3 Distance-based Analysis

We also compare the performance of the two models on relations based on the distance between the entities in a relation in Table 4.6. We find that the performance of both the models is very low for distance greater than 7. SPTree model can identify 36 relations

out of 131 such relations correctly, while our model can only identify 20 relations in this category. We manually compare the output of the two systems on these cases on several examples to understand the gain of using dependency tree on longer distances. Interestingly, the majority of these relations belong to PHYS type, thus resulting in lower performance on PHYS as discussed in Section 4.6.2. We found that there were a few instances of co-reference errors as shown in S2 in Table 4.5. Our model identifies a PHYS relation between "here" and "baghdad", whereas the gold annotation has PHYS relation between "location" and "baghdad". We think that incorporating these co-reference information during both training and evaluation will further improve the performance of both systems. Another source of error that we found was the inability of our system to extract entities (lower recall) as in S3. Our model could not identify the FAC entity "residence". Hence, we think an improvement on entity performance via methods like pretraining might be helpful in identifying more relations. For distance less than 7, we find that our model has better recall but lower precision, as expected.

4.7 Chapter Summary

In the work describe in this chapter, we propose a novel attention-based LSTM model for joint extraction of entity mentions and relations. Experimentally, we found that our model significantly outperforms feature-rich structured perceptron joint model by Li and Ji (2014). We also compare our model to an end-to-end LSTM model by Miwa and Bansal (2016a) which comprises of a sequence layer for entity extraction and a tree-based dependency layer for relation classification. We find that our model, without access to dependency trees, POS tags, etc performs within 1% on entities and 2% on relations on ACE05 dataset. We also find that our model performs significantly better than their tree-based model on the ART relation, while their tree-based model performs

better on PHYS and PART-WHOLE relations; the two models perform comparably on all other relation types.

Some of the immediate approaches to improve this model would be to explore pre-training methods which were shown to improve recall on entity and relation performance by Miwa and Bansal (2016a). In this work, we also introduce bi-directional output encoding as well as an objective to learn multiple relations. However, this presents the challenge of combining predictions from the two directions. We use heuristics in this work to combine the predictions. We think that using probabilistic methods to combine model predictions from both directions may further improve the performance. Also, using Sparsemax (Martins and Astudillo, 2016) instead of Softmax for multiple relations, as the former is more suitable for multi-label classification for sparse labels.

CHAPTER 5

NESTED NAMED ENTITIES

In Chapter 3, we made an assumption that the entity mentions and relations are not overlapping. In order to achieve that, we preprocessed our datasets to only keep the embedded entity mentions similar to the previous work (Yang and Cardie, 2013). Similarly, in Chapter 4, we only extract the head of the entity mentions instead of the full text of entity mentions. As a result, we remove all overlapping entity mentions in both these previous works resulting in the limitation that we are unable to identify both these overlapping mentions and their relations. In this chapter, we develop models that are capable to extracting nested entity mentions and hence our models for extracting facts and opinions need not depend on these preprocessing steps which results in loss of information. The work presented in this chapter is based on Katiyar and Cardie (2018). We first describe the task and then present neural models to extract nested entity mentions.

Named entity recognition (or named entity detection) is the task of identifying text spans associated with proper names and classifying them according to their semantic class such as person, organization, etc. It is related to the task of *mention detection* (or entity mention recognition) in which text spans referring to named, nominal or prominal entities are identified and classified according to their semantic class (Florian et al., 2004). Both named entity recognition and entity mention detection are fundamental components in information extraction systems: several downstream tasks such as relation extraction (Mintz et al., 2009), coreference resolution (Chang et al., 2013) and fine-grained opinion mining (Choi et al., 2006) rely on both.

Many approaches have been successfully employed for the tasks of named entity recognition and mention detection, including linear-chain conditional random fields

(Lafferty et al., 2001) and semi-Markov conditional random fields (Sarawagi and Cohen, 2005). However, most such methods suffer from an inability to handle *nested* named entities, *nested* entity mentions, or both. As a result, the downstream tasks necessarily ignore these nested entities along with any semantic relations among them. Consider, for example, the excerpts below:

(S1) Employing the [EBV - transformed [human B cell line]$_{\text{CELL\_LINE}}$]$_{\text{CELL\_LINE}}$ SKW6.4, we demonstrate . . .

(S2) . . . [the burial site of [Sheikh Abbad]$_{\text{PERSON}}$]$_{\text{LOCATION}}$ is located . . .

S1 shows a nested named entity from the GENIA dataset (Ohta et al., 2002): "human B cell line" and "EBV - transformed human B cell line" are both considered named entities of type CELL_LINE where the former is embedded inside the latter. S2, derived from the ACE corpora[1], shows a PERSON named entity ("Sheikh Abbad") nested in an entity mention of type LOCATION ("the burial site of Sheikh Abbad"). Most existing methods for named entity recognition and entity mention detection would miss the nested entity in each sentence.

Unfortunately, nested entities can be fairly common: 17% of the entities in the GENIA corpus are embedded within another entity; in the ACE corpora, 30% of sentences contain nested named entities or entity mentions, thus warranting the development of efficient models to effectively handle these linguistic phenomena.

Feature-based methods are the most common among those proposed for handling nested named entity and entity mention recognition. Alex et al. (2007), for example, proposed a cascaded CRF model but it does not identify nested named entities of the same type. Finkel and Manning (2009) proposed building a constituency parser with

[1] `https://catalog.ldc.upenn.edu/LDC2005T09` (ACE2004) and `https://catalog.ldc.upenn.edu/LDC2006T06` (ACE2005)

constituents for each named entity in a sentence. Their approach is expensive, i.e., time complexity is cubic in the number of words in the sentence. Lu and Roth (2015) later proposed a mention hypergraph model for nested entity detection with linear time complexity. And recently, Muis and Lu (2017) introduced a multigraph representation based on mention separators for this task. All of these models depend on manually crafted features. In addition, they cannot be directly applied to extend current state-of-the-art recurrent neural network-based models — for flat named entity recognition (Lample et al., 2016) or the joint extraction of entities and relations (Katiyar and Cardie, 2016) — to handle nested entities.

In this chapter, we propose a recurrent neural network-based model for nested named entity and nested entity mention recognition. We present a modification to the standard LSTM-based sequence labeling model (Sutskever et al., 2014) that handles both problems and operates linearly in the number of tokens and the number of possible output labels at any token. The proposed neural network approach additionally jointly models entity mention *head*[2] information, a subtask found to be useful for many information extraction applications. Our model significantly outperforms the previously mentioned hypergraph model of Lu and Roth (2015) and Muis and Lu (2017) on entity mention recognition for the ACE2004 and ACE2005 corpora. It also outperforms their model on *joint* extraction of nested entity mentions and their heads. Finally, we evaluate our approach on nested named entity recognition using the GENIA dataset and show that our model outperforms the previous state-of-the-art parser-based approach of Finkel and Manning (2009).

In the sections that follow, we describe related work (Section 6.1); our encoding scheme (Section 5.2); our bi-directional recurrent model (Section 6.2); training (Section 6.3); the experiments on ACE dataset and GENIA corpus (Section 6.4); error anal-

[2]This involves identifying the headword of a named entity or entity mention.

ysis (Section 5.6); and conclusion and future work (Section 5.7).

5.1 Related Work

Several methods have been proposed for named entity recognition in the existing literature as summarized by Nadeau and Sekine (2007) in their survey paper. Early techniques in the supervised domain have been based on hidden markov models (e.g., Zhou and Su (2002)) or, later, conditional random fields (CRFs) (e.g., McDonald and Pereira (2005)).

Many fewer approaches, however, have addressed the problem of nested entities. Alex et al. (2007) presented several techniques based on CRFs for nested named entity recognition for the GENIA dataset. They obtained their best results from a cascaded approach, where they applied CRFs in a specific order on the entity types, such that each CRF utilizes the output derived from previous CRFs. Their approach could not identify nested entities of the same type. Finkel and Manning (2009) proposed a CRF-based constituency parser for nested named entities such that each named entity is a constituent in the parse tree. Their model achieved state-of-the-art results on the GENIA dataset. However, the time complexity of their model is $O(n^3)$, where n is the number of tokens in the sentence, making inference slow. As a result, we do not adopt their parse tree-based representation of nested entities and propose instead a linear time directed hypergraph-based model similar to that of Lu and Roth (2015). Directed hypergraphs were also introduced for parsing by Klein and Manning (2001).

While most previous efforts for nested entity recognition were limited to named entities, Lu and Roth (2015) addressed the problem of nested entity mention detection where mentions can either be named, nominal or pronominal. Their hypergraph-based approach is able to represent the potentially exponentially many combinations of nested

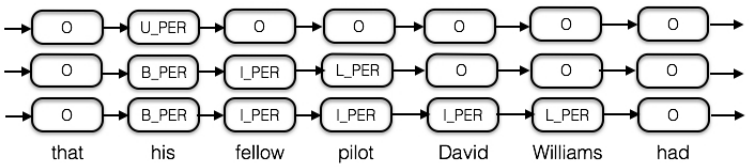


Figure 5.1: Nested entity mentions in an unfolded hypergraph. Each row corresponds to an entity mention sequence using the well known B_ (beginning of mention), I_ (inside a mention), L_ (last token of an entity mention), O (outside any entity mention), U_ (a single-token entity mention) tagging scheme.

mentions of different types. They adopted a CRF-like log-linear approach to learn these mention hypergraphs and employed several hand-crafted features defined over the input sentence and the output hypergraph structure. Our approach also learns a similar hypergraph representation with differences in the types of nodes and edges in the hypergraph. It does not depend on any manually crafted features. Also, our model learns the hypergraph greedily and significantly outperforms their approach.

Recently, Muis and Lu (2017) introduced the notion of mention separators for nested entity mention detection. In contrast to the hypergraph representation that we and Lu and Roth (2015) adopt, they learn a multigraph representation and are able to perform exact inference on their structure. It is an interesting orthogonal possible approach for nested entity mention detection. However, we will show that our model also outperforms their approach on all tasks.

Recently, recurrent neural networks (RNNs) have been widely applied to several sequence labeling tasks achieving state-of-the-art results. Lample et al. (2016) proposed neural models based on long short term memory networks (LSTMs) and CRFs for named entity recognition and another transition-based approach inspired by shift-reduce parsers. Both models achieve performance comparable to a state-of-the-art model (Luo et al., 2015), but neither handles nested named entities.

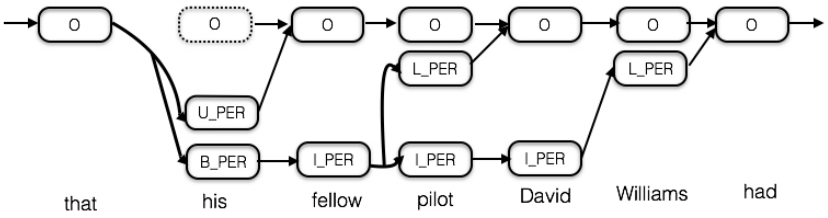


Figure 5.2: Directed hypergraph constructed for the example shown in Figure 5.1. Curved edges represent hyperarcs and straight edges are normal edges.

5.2 Encoding Scheme

Figure 5.1 shows the desired sequence tagging output for each of three overlapping PER entities ("his", "his fellow pilot" and "his fellow pilot David Williams") according to the standard BILOU tag scheme. Our approach relies on the fact that we can (1) represent these three tag sequences in the single hypergraph structure of Figure 5.2 and then (2) design an LSTM-based neural network that produces the correct nested entity hypergraph for a given input sentence. In the paragraphs just below we provide a general description of hypergraphs and our task-specific use of them. Sections 3.1 and 3.2 describe the hypergraph construction process; Section 4 presents the LSTM-based sequence tagging method for automating hypergraph construction.

We express our structured prediction problem such that it corresponds to building a hypergraph that encodes the token-level gold labels for all entities in the input sentence.[3] In particular, we represent the problem as a directed hypergraph. For those new to this formalism, directed hypergraphs are very much like standard directed graphs except that nodes are connected by hyperarcs that connect a *set* of tail nodes to a *set* of head nodes. To better explain our desired output structure, we further distinguish between two types of hyperarcs — *normal edges* (or arcs) that connect a single tail node to a single head

[3]We note that the complete hypergraph for the example in Figure 5.1 would include nodes for all possible label types at each timestep and all possible hyperarcs between them. In this work, however, we only greedily build a sub-hypergraph for the gold labels when training.

node, and *hyperarcs* that contain more than one node either as the head or as the tail. The former are shown as straight lines in Figure 5.2; the latter as curved edges.

In our encoding of nested entities, a hyperarc is introduced when two or more entity mentions requiring different label types are present at the same position. In Figure 5.2, for example, the nodes "O" (corresponding to the input token "that") and the nodes "U_PER" and "B_PER" (corresponding to the input token "his") are connected by a hyperarc because three entity mentions start at this time step from the tail "O" node (two of which share the "B_PER" tag).[4]

5.2.1 Hypergraph Construction

Let us first discuss how the problem of nested entity recognition can be expressed as finding a hypergraph. Our goal is to represent the BILOU tag sequences associated with "his", "his fellow pilot" and "his fellow pilot David Williams" as the single hypergraph structure of Figure 5.2. This is accomplished by collapsing the shared states (labels) in the output entity label sequences into a single state as shown in Figure 5.2: e.g., the three "O" labels for "that" become a single "O"; the two "B_PER" labels at "his" are collapsed into one "B_PER" node that joins "U_PER", the latter of which represents the entity mention "his". Thus at any time step, the representation size is bounded by the number of possible output states instead of the potentially exponential number of output sequences. We then also adjust the directed edges such that they have the same type of head node and the same type of tail node as before in Figure 5.1.

If we look closely at Figure 5.2 then we realise that there is an extra "O" node in the

[4]In contrast, note that the nodes "L_PER" and "O" corresponding to the input token "pilot" and the node "O" corresponding to the token "David" are connected by normal edges. Hence, our hypergraph structure contains only one special kind of hyperarc which connects a single tail node to multiple head nodes. We do not have hyperarcs that connect multiple tail nodes to a single head node.

hypergraph corresponding to the token "his" which did not appear in any entity output sequence in Figure 5.1: in our task-specific hypergraph construction we make sure that there is an "O" node at every timestep to model the possibility of beginning of a new entity. The need for this will become more clear in Section 6.2.

Note that the hypergraph representation of our model is similar to Lu and Roth (2015). Also, the expressiveness of our model is exactly the same as Lu and Roth (2015); Muis and Lu (2017). The major difference in the two approaches is in learning.

5.2.2 Edge Probability

In this section, we discuss our assignment of probabilities to all the possible edges from a tail node which helps in the greedy construction of the hypergraph. Thus at any timestep t, let g_{t-1} be the tail node and x be the current word of the sentence; then we model probability distribution over all the possible types of head nodes (different output tag types) conditioned on the tail node and the current word token. In our work we use hidden representations learned from an LSTM model as features to learn these probability distributions using a cross-entropy objective.

It is important to note that there are two types of directed edges in this hypergraph – simple edges for which there is only one head node for every tail node which can be learned as in a traditional sequence labeling task, or hyperarcs that connect more than one head node to a tail node. We learn the set of head nodes connected to a tail node by expressing it as a multi-label learning problem as described in Section 6.3.

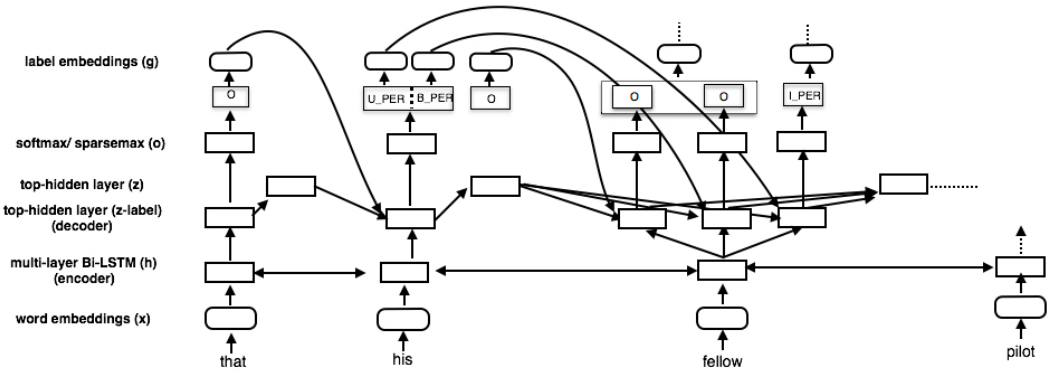


Figure 5.3: Dynamically computed network structure based on bi-LSTMs for nested entity mention extraction. We show part of the structure for the entity mentions in the running example in Figure 5.1.

5.2.3 Extracting Entity Mentions

As described in Section 5.2.2, we can assign probabilities to the different types of edges in the hypergraph and at the time of decoding we choose for each token the (normal) edge(s) with maximum probability and the hyperarcs with probability above a predefined threshold. Thus, we can extract edges at the time of decoding. Ultimately, however, we are interested in extracting nested entities from the hypergraph. For this, we construct an adjacency matrix from the edges discovered and perform depth-first search from the sentence-initial token to discover the entity mentions. This is described in detail in Section 5.4.1.

5.3 Method

We use a standard LSTM-based sequence labeling model to learn the nested entity hypergraph structure for an input sentence. Figure 6.2 shows part of the network structure. It is a standard bi-directional LSTM network except for a difference in the top hidden

layer. When computing the representation of the top hidden layer L at any time step t, in addition to making use of the hidden unit representation from the previous time step $t-1$ and hidden unit representation from the preceding layer $L-1$, we also input the label embedding of the gold labels from the previous time step. For the token "fellow" in Figure 6.2, for example, we compute three different top hidden layer representations, conditioned respectively on the three labels "U_PER", "B_PER" and "O" from the previous time step $t-1$. Thus, we can model complex interactions between the input and the output. Before passing the learned hidden representation to the next time step, we average the three different top hidden layer representations. In this manner, we can model the interactions between the different overlapping labels and also it is computationally less expensive than storing the hidden layer representations for each label sequence.

5.3.1 Multi-layer Bi-LSTM

We use a multi-layer bi-directional LSTM encoder, for its strength in capturing long-range dependencies between tokens, a useful property for information extraction tasks.

Using LSTMs, we can compute the hidden state $\overrightarrow{\boldsymbol{h}}_t$ in the forward direction and $\overleftarrow{\boldsymbol{h}}_t$ in the backward direction for every token, and use a linear combination of them as the token representation:

$$\overrightarrow{\boldsymbol{h}}_t^{(l)} = \text{LSTM}(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}}_{t-1}) \tag{5.1}$$

$$\overleftarrow{\boldsymbol{h}}_t^{(l)} = \text{LSTM}(\boldsymbol{x}_t, \overleftarrow{\boldsymbol{h}}_{t+1}) \tag{5.2}$$

$$\boldsymbol{z}_t^{(l)} = \overrightarrow{\boldsymbol{V}}\overrightarrow{\boldsymbol{h}}_t^{(l)} + \overleftarrow{\boldsymbol{V}}\overleftarrow{\boldsymbol{h}}_t^{(l)} + \boldsymbol{b}^l \tag{5.3}$$

5.3.2 Top Hidden Layer

At the top hidden layer, we have a decoder-style model, with a crucial twist to accommodate the hypergraph structure, which may have multiple gold labels at the previous step. At each token t and *for each* gold label at the previous step g^k_{t-1}, our network takes the hidden representation from the previous layer $\boldsymbol{z}^{(L-1)}_t$, the hidden decoder state $\boldsymbol{h}^{(L)}_{t-1}$, as well as the gold label embedding $\boldsymbol{g}^k_{t-1}$ from the previous time step, and computes:

$$\boldsymbol{h}^{(L),k}_t = \text{LSTM}(\boldsymbol{z}^{(L-1)}_t, \boldsymbol{h}^{(L)}_{t-1}, \boldsymbol{g}^k_{t-1}) \tag{5.4}$$

Unlike the encoder LSTM, this decoder LSTM is single-directional and bifurcates when multiple gold labels are present. We use the decoder hidden states $\boldsymbol{h}^{(L),k}_t$ in the output layer for prediction, as explained in Section 5.3.3. However, before passing the hidden representation to the next time step we average $\boldsymbol{h}^{(L),k}_t$ over all the gold labels k:

$$\boldsymbol{h}^{(L)}_t = \frac{1}{|G_{t-1}|} \sum_k \boldsymbol{h}^{(L),k}_t \tag{5.5}$$

Thus, $\boldsymbol{h}^{(L)}_t$ summarizes the information for all the gold labels from the previous time step.

5.3.3 Entity Extraction

For each token t and previous gold label g^k_{t-1}, we use the decoder state $\boldsymbol{h}^{(L),k}_t$ to predict a *probability distribution* over the possible candidate labels using a linear layer followed by a normalizing transform (illustrated below with softmax). The outputs can be inter-

	ACE2004			ACE2005		
Method	P	R	F1	P	R	F1
MH-F (Lu and Roth, 2015)	70.0	59.2	63.8	70.0	56.9	62.8
Muis and Lu (2017)	72.7	58.0	64.5	69.1	58.1	63.1
LSTM-flat	70.3	48.4	57.3	62.4	49.4	55.1
LSTM-output_layer	72.0	63.3	67.4	66.3	68.2	67.2
Our model (softmax)	72.2	65.2	68.5	70.1	67.9	69.0
Our model (sparsemax)	**73.6**	**71.8**	**72.7**	**70.6**	**70.4**	**70.5**

Table 5.1: Performance on ACE2004 and ACE2005 test set on mention extraction and classification.

preted as conditional probabilities for the next label given the current gold label:

$$\boldsymbol{o}_t^k = \boldsymbol{U}\boldsymbol{h}_t^{(L),k} + \boldsymbol{b} \tag{5.6}$$

$$\hat{\boldsymbol{e}}_t^k = \text{softmax}(\boldsymbol{o}_t^k) \tag{5.7}$$

$$p(y_t = c | y_{t-1} = g_{t-1}^k) = (\boldsymbol{e}_t^k)_c \tag{5.8}$$

Special care is required, however, since the desired output has hyperarcs. As shown in Figure 5.2, there is an hyperarc between "I_PER" corresponding to the token "fellow" and the label set "L_PER" and "I_PER" corresponding to the token "pilot". Thus, in our network structure conditioned on the previous label "I_PER" in this case, we would like to predict both "L_PER" and "I_PER" as the next labels. To accommodate this, we use a multi-label training objective, as described in Section 6.3.

5.4 Training

We train our model using two different multi-label learning objectives. The idea is to represent the gold labels as a distribution over all possible labels, encoded as a vector $\boldsymbol{e}$. Hence, for simple edges, the distribution has a probability of 1 for the unique gold label ($\boldsymbol{e}_g = 1$), and 0 everywhere else. For hyperarcs, we distribute the probability mass

uniformly over all the gold labels in the gold label set ($\boldsymbol{e}_g^k = \frac{1}{|G|}$ for all k). Thus, for the example described earlier in Section 5.3.3, both the labels "L_PER" and "I_PER" receive a probability of 0.5 in the gold label distribution $\boldsymbol{e}_t^k$, conditioned on the label "I_PER" from the previous time step.

Softmax. Our first training method uses softmax to estimate the predicted probabilities, and the KL-divergence multi-label loss between the true distribution $\boldsymbol{e}_t^k$ and the predicted distribution $\hat{\boldsymbol{e}}_t^k = \text{softmax}(\boldsymbol{o}_t^k)$:

$$\ell_{t(\text{softmax})}^k = -\sum_c \left(\boldsymbol{e}_t^k\right)_c \log\left(\hat{\boldsymbol{e}}_t^k\right)_c \tag{5.9}$$

Sparsemax. Our second training method makes use of *sparsemax*, recently introduced by Martins and Astudillo (2016) as a sparse drop-in replacement to softmax, as well as a loss function. Unlike softmax, which always outputs a nonzero probability for any output, sparsemax outputs zero probability for most of the unlikely classes, leading to good empirical results on multi-label tasks. For our problem, there are only a few nested entities at any timestep in the gold labels thus using a training objective that learns a sparse distribution is more appropriate. Sparsemax can be used to filter part of the output space as in the case for multi-label problems thus leaving non-zero probability on the desired output labels.

Formally, sparsemax returns the euclidean projection of its input $\boldsymbol{o}$ onto the probability simplex:

$$\hat{\boldsymbol{e}} = \text{sparsemax}(\boldsymbol{o}) := \underset{\hat{\boldsymbol{e}} \in \Delta}{\text{argmin}} \, \|\boldsymbol{o} - \hat{\boldsymbol{e}}\|^2 \tag{5.10}$$

The corresponding loss, a sparse version of the KL divergence, is (up to a constant):

$$\ell_{t(\text{sparsemax})}^k = -2\boldsymbol{e}_t^{k\top}\boldsymbol{o}_t^k + \sum_{c:(\hat{\boldsymbol{e}}_t^k)_c \neq 0} \left((\boldsymbol{o}_t^k)_c^2 - \tau^2\right) \tag{5.11}$$

This function is convex and differentiable, and the quantity τ is a biproduct of the simplex projection, as described in Martins and Astudillo (2016).

For either choice of probability estimation, the total loss of a training sample is the sum of losses for each token and for each previous gold label:

$$\mathcal{L} = \sum_{t} \sum_{k \in G_{t-1}} \ell_t^k. \tag{5.12}$$

5.4.1 Decoding

At the time of inference, we greedily decode our hypergraph from left-to-right to find the most likely sub-hypergraph. During training, at each timestep the most likely label set is learned conditioned on a gold label from the previous timestep. However, gold labels are not available at test time. Thus, we use the predicted labels from the previous time step as an input to the current time step to find the most likely label set. We use a hard threshold T to determine the predicted label set $P_t^k = \{c : \left(\hat{\boldsymbol{e}}_t^k\right)_c > T\}$

We can get the most likely label set P_t^c for any predicted label at the previous time step $c \in P_{t-1}^k$ using the above decoding strategy. We now combine these inferences to find the most likely entity mention sequences. We construct an adjacency matrix $\boldsymbol{A}$ for each time step, such that $\boldsymbol{A}[\hat{\boldsymbol{e}}_{t-1}^c][\hat{\boldsymbol{e}}_t^k]$ += 1 for every c in the predicted label set P_t^k at timestep t conditioned on $\hat{\boldsymbol{e}}_t^k$ and for every k in predicted labels P_{t-1} at time step $t-1$. This can be viewed as a directed hypergraph with several connected components. We then perform a depth-first search on this directed hypergraph to find all the entity mentions in the sentence.

	ACE2004			ACE2005		
Method	P	R	F1	P	R	F1
MH-F (Lu and Roth, 2015)	74.4	50.0	59.8	63.4	53.8	58.3
Our model(softmax)	68.2	60.5	64.2	67.5	62.3	64.8
Our model(sparsemax)	**72.3**	**66.8**	**69.7**	**70.6**	**69.8**	**70.2**

Table 5.2: Performance on ACE2004 and ACE2005 test set on joint entity mention and its head prediction. Muis and Lu (2017) do not predict head of the nested entity mentions.

5.4.2 Modeling Entity Heads for ACE datasets

The ACE datasets also have annotations for mention heads along with the entity mentions. For example, a sentence with the entity mention "the U.S. embassy" also contains an annotation for its head word which is "embassy" in this case. Thus, we modify our model to also extract the head of the entity mentions for ACE dataset. We jointly model the entity mentions and their heads. To do this, we propose a simple extension to our model by only changing the output label sequence. We introduce new labels starting with "H" to indicate that the current token in the entity mention is part of its head. Thus, we only change the output label sequence for the entity mentions to include the head label: We train with the label sequence "B_ORG I_ORG H_ORG" instead of "B_ORG I_ORG L_ORG". Also, for all our entity sequences we predict the "O" tag at the end, hence we can still extract the entity mentions. At decoding time, we output the sequence of words with the "H" tag as the head words for a mention.

5.5 Experiments

We evaluate our model on two tasks – nested entity mention detection for the ACE corpora and nested named entity recognition for the GENIA dataset.

5.5.1 ACE Experiments

In this section, we present the ACE dataset used for evaluating the performance of our proposed models. We also describe the previous models and compare the performance of our models with them.

5.5.1.1 Data

We perform experiments on the English section of the ACE2004 and ACE2005 corpora. There are 7 main entity types — Person (PER), Organization (ORG), Geographical Entities (GPE), Location (LOC), Facility (FAC), Weapon (WEA) and Vehicle (VEH). For each entity type, there are annotations for the entity mention and mention heads.

5.5.1.2 Evaluation Metrics

We use a strict evaluation metric similar to Lu and Roth (2015): an entity mention is considered correct if both the mention span and the mention type are exactly correct. Similarly, for the task of joint extraction of entity mentions and mention heads, the mention span, head span and the entity type should all exactly match the gold label.

5.5.1.3 Baselines and Previous Models

We compare our model with the feature-based model (MH-F) on hypergraph structure (Lu and Roth, 2015) on both entity mention detection as well as the joint mention and mention heads extraction. We also compare with Muis and Lu (2017) on entity mention detection only as their model cannot detect head phrases of the entity mentions. Lu and Roth (2015) compare their approach with CRF-based approaches such as a linear-chain

CRF, semi-markov CRF and a cascaded approach (Alex et al., 2007) and show that their model outperforms them. Hence, we do not include those results in this chapter.

We also implement several LSTM-based baselines for comparison. Our first baseline is a standard sequence labeling LSTM model (LSTM-flat). A sequence model is not capable of handling the nested mentions, so we remove the embedded entity mention and keep the mention longer in length. Our second baseline is a hypergraph model (LSTM-output_layer) except that the dependencies are only modeled at the output layer and hence there are no connections to the top-hidden layer from the label embeddings from the previous timestep; instead, these connections are limited to the output layer.

5.5.1.4 Hyperparameters and Training Details

We use Adadelta (Zeiler, 2012) for training our models. We initialize our word vectors with 300-dimensional word2vec (Mikolov et al., 2013) word embeddings. These word embeddings are tuned during training. We regularize our network using dropout (Srivastava et al., 2014), with the dropout rate tuned on the development set. There are 3 hidden layers in our network and the dimensionality of hidden units is 100 in all our experiments. And we set the threshold T as 0.3.

5.5.1.5 Results

We show the performance of our approaches in Table 5.1 compared to the previous state-of-the-art system (Lu and Roth, 2015; Muis and Lu, 2017) on both the ACE2004 and ACE2005 datasets. We find that our LSTM-flat baseline that ignores embedded entity mentions during training performs worse than Lu and Roth (2015); however, our other neural network-based approaches all outperform the previous feature-based approach.

Among the neural network-based models, we find that our models that construct a hypergraph perform better than the LSTM-flat models. Also, our approach that models dependencies between the input and the output by passing the prediction from the previous timestep as shown in Figure 6.2 performs better than the LSTM-output_layer model which only models dependencies at the output layer. Also, as expected, the sparsemax method that produces a sparse probability distribution performs better than the softmax approach for modeling hyperedges. In summary, our sparsemax model is the best performing model.

Joint Modeling of Heads We report the performance of our best performing models on the joint modeling of entity mentions and its head in Table 5.2. We show that our sparsemax model is still the best performing model. We also find that as the total number of possible labels at any timestep increases because of the way we implemented the entity heads, the gains that we get after incorporating sparsemax are significantly higher compared to the results shown in Table 5.1.

5.5.2 GENIA Experiments

In this section, we present the GENIA dataset used for evaluating the performance of our proposed models. We also describe the previous models and compare the performance of our models with them.

5.5.2.1 Data

We also evaluate our model on the GENIA dataset (Ohta et al., 2002) for nested named entity recognition. We follow the same dataset split as Finkel and Manning (2009); Lu

Method	P	R	F1
Finkel and Manning (2009)	75.4	65.9	70.3
MH-F (Lu and Roth, 2015)	72.5	65.2	68.7
Muis and Lu (2017)	75.4	66.8	70.8
LSTM-flat	75.5	63.5	68.9
LSTM-output_layer	78.4	67.9	72.8
Our model (softmax)	76.7	**71.1**	**73.8**
Our model (sparsemax)	**79.8**	68.2	73.6

Table 5.3: Performance on the GENIA dataset on nested named entity recognition.

and Roth (2015); Muis and Lu (2017). Thus, the first 90% of the sentences were used in training and the remaining 10% were used for evaluation. We also consider five entity types – DNA, RNA, protein, cell line and cell type.

5.5.2.2 Baselines and Previous Models

We compare our model with Finkel and Manning (2009) based on a constituency CRF-based parser and the mention hypergraph model by Lu and Roth (2015) and a recent multigraph model by Muis and Lu (2017).

5.5.2.3 Results

Table 5.3 shows the performance of our different models compared to the previous models. Interestingly, our LSTM-flat model outperforms Lu and Roth (2015). We suspect that it is because we use pretrained word embeddings[5] trained on PubMed data (Pyysalo et al., 2013) whereas Lu and Roth (2015) did not have access to them. We again find that our neural network model outperforms the previous state-of-the-art (Finkel and Man-

[5]Word vectors trained on PubMed data are available at `http://bio.nlplab.org/#source-data`.

ning, 2009; Muis and Lu, 2017) system. However, we see that both softmax and sparsemax models perform comparably on this dataset.

5.6 Error Analysis

Consistent with existing results on the joint modeling of related tasks in NLP, we find that joint modeling of heads and their entity mentions leads to an increase in F-score by 1pt (i.e., 71.4 for the sparsemax model on the ACE2005 dataset) on the performance of the entity mentions. The precision on extracting entity mentions is 72.1 (vs. 70.6 in Table 5.1) for our sparsemax model for the ACE2005 dataset.

Example S1 below compares the output from a softmax vs. a sparsemax model on the joint modeling of an entity mention and its head on the ACE2005 dataset. Gold-standard annotations are shown in *red*.

(S1) [[[They]]]$_{\text{PERSON}}$ don't abandon [[[[[their]]]]$_{\text{PERSON}}$ patients]]$_{\text{PERSON}}$, except for the high premiums of a few specialities?

Based on the gold standard, the models are required to extract "their" — an entity mention of type PER as well as its head — and "their patients", which overlaps with the previous entity mention "their" and has the head word "patients". This means that the models are required to predict a hyperedge from "O" to "H_PER; B_PER". We find that the softmax model shown in *blue* can only predict the entity mention "their" omitting completely the entity mention "their patients" whereas the sparsemax model shown in *green* can predict both nested entities. Overall then, sparsemax seems to allow the modeling of hyperedges more efficiently compared to the softmax model and performance gains are due to extracting more nested entities with the help of sparsemax model.

5.6.1 Limitations and Future Directions

We also manually scanned the test set predictions on ACE dataset for our sparsemax model to understand its current limitations.

Document Context. Given the following sentence

(S2) [They]$_{\text{VEHICLE}}$ roar, [they]$_{\text{VEHICLE}}$ screech.

the sparsemax model predicts both entity mentions of "they" as PER entity type. Only if the previous sentence in the corpus is accessible — "And if you ride inside that tank, it is like riding in the bowels of a dragon" — can we understand that "they" in S2 refers to the tank and hence is a VEH. Thus, our model can be improved by providing additional context for each sentence rather than making predictions on each sentence in the corpus independently.

Pronominal Entity Mention (It). Next, consider examples S3 and S4:

(S3) [It]$_{\text{FACILITY}}$ also seemed to be [some kind of monitoring station]$_{\text{FACILITY}}$.

(S4) It does not matter to [these people]$_{\text{PERSON}}$ that crime has skyrocketed . . .

In the example sentences, "It" refers to a facility and an event, respectively. Our model does not distinguish between the two cases and always predicts the token "It" as a non-entity. We found this true for all occurrences of the token "It" in our test set. The incorporation of coreference information can potentially overcome this limitation.

Inconsistency in Gold-standard Annotations. We also identified potential inconsistencies in the gold-standard annotations.

(S5) ...results may affect what happens to [both of these teams]$_{\text{ORG}}$, but in just ...

For S5, the gold-standard annotation for "both of these teams" is an ORG entity mention with the token "teams" as its head word. Our sparsemax model identifies the entity mention correctly but instead predicts the token "both" as the head. It also identifies "these teams" as another nested entity mention with the head word "teams". In contrast, however, we also found entity mentions such as "all of the victims that get a little money" for which the gold-standard has "all" annotated as its head and another nested mention "the victims that get a little money" with "victims" as the head. We recognize this as an inconsistency in the gold-standard annotation.

5.7 Chapter Summary

In the work described in this chapter, we present a novel recurrent network-based model for nested named entity recognition and nested entity mention detection. We propose a hypergraph representation for this problem and learn the structure using an LSTM network in a greedy manner. We show that our model significantly outperforms a feature based mention hypergraph model (Lu and Roth, 2015) and a recent multigraph model (Muis and Lu, 2017) on the ACE dataset. Our model also outperforms the constituency parser-based approach of Finkel and Manning (2009) on the GENIA dataset.

However, the work described in the chapter is limited to making local decisions, thus it would be interesting to learn global dependencies between the output labels for such a hypergraph structure and training the model globally. Also, we only focus our work

on hypergraph representation but it would be interesting to also experiment with different representations such as the one in Finkel and Manning (2009) and use the recent advances in neural network approaches (Vinyals et al., 2015b) to learn the constituency parse tree efficiently.

CHAPTER 6

DOCUMENT-LEVEL CONTEXT IN RELATION EXTRACTION

In this chapter, we present our approach to extend relation extraction to the document-level. As described in Chapter 2, information extraction entails entity extraction for extraction of all mentions of entities, coreference resolution for merging the mentions of the same entities, entity classification for identifying the type of the entities and relation extraction for identifying the relationships among the entities. In our work described so far in this thesis, we have only focused on sentence-level information extraction for simplicity and mostly ignored coreference resolution for identifying all mentions of an entity. As a result, we are only able to recover relations between mentions of entities instead of the entities itself. In this chapter, we will extend our previous models to also incorporate this crucial missing component in information extraction and investigate its effect on relation extraction. We will first motivate the problem by introducing some of the current approaches and then we will present our approach.

Traditionally, the task of relation extraction involves determining the relations that exist between known typed entity mentions. Notably, this task is most often constrained to the *sentence-level*, i.e. to finding relations between two entity mentions present in the same sentence. As a result, a lot of the neural network approaches (Nguyen and Grishman, 2015; Miwa and Bansal, 2016a; Katiyar and Cardie, 2017; Zhang et al., 2017b; Christopoulou et al., 2018) to this problem are also sentence-based: they generally focus on learning a relation classifier on all pairs of mentions independently in each sentence. In recent work, Christopoulou et al. (2018) model interactions among the entity mentions. Nonetheless, it still fundamentally operates at the sentence-level.

Among the few systems that incorporate features beyond sentence-level for relation extraction is the feature-engineered system by Chan and Roth (2010). They show im-

provement in relation extraction by using gold-standard coreference relations among entity mentions as an additional constraint in their Integer Linear Programming framework. In more recent neural network approaches, Sanh et al. (2019) and Luan et al. (2018, 2019) propose a multi-task approach for entity extraction, relation extraction and coreference resolution jointly. They show that simultaneously learning coreference resolution using an external corpus improves both entity and relation extraction. However, when modeling these tasks together it is difficult to isolate the effect of coreference on only relation extraction. In fact, Luan et al. (2019) found that ablating coreference hurts entity extraction but improves relation extraction on ACE05 corpus. Thus, to the best of our knowledge, there is no neural network based system that studies the direct effect of coreference on relation extraction and show improvements similar to Chan and Roth (2010). In this work, we investigate the use of discourse-level context in the form of coreference relations among entity mentions for neural network based relation extraction.

Consider, for example, the snippet of text in Figure 6.1. "North Korea" and "it" refers to the same entity and hence no other interesting relation exists between them. Moreover, coreference based entity-level information might allow a relation extraction model to find additional instances of relations already identified in the text. For example, in Figure 6.1, we see that "They" is also coreferent with "North Korea" and "it", and that the two occurrences of "weapons" are coreferent such that the relations between these mentions are two instances of the same relation. As a result, we propose to augment neural relation extraction systems with explicit entity-level representations that incorporate coreference information.

In this work, we extend the relation extraction model by Katiyar and Cardie (2017) because it does not require any external syntactic features to (1) include a sentence-

Figure 6.1: An example document from the ACE05 dataset showing some of the mentions of entities and relations of interest. The different entity mentions for an entity are indicated by the same color. The relations of interest are indicated by curved lines.

level *recency* feature that indicates the distance between entity mentions, and (2) include document-level noun phrase coreference information encoded as *entity-level features*. In the experiments on the ACE05 dataset, we improve the state-of-the-art (Miwa and Bansal, 2016a) F1 score by 2 points by incorporating a recency bias into the position-aware attention model (Zhang et al., 2017b); our entity-level representations further improve F1 score by 1 point.

6.1 Related Work

Relation Extraction. Relation extraction (Zhou et al., 2005) has been widely studied both as a separate task (Bunescu and Mooney, 2005; Zelenko et al., 2003; Socher et al., 2012; dos Santos et al., 2015; Xu et al., 2015a,b) as well as a joint task with entity

mention detection (Miwa and Sasaki, 2014; Li and Ji, 2014; Miwa and Bansal, 2016a; Gupta et al., 2016; Katiyar and Cardie, 2017; Zhang et al., 2017a) using both feature-based as well as neural network models. However, it is known that the joint extraction of entities and relations benefits both the tasks (Li and Ji, 2014).

Since neural network models became popular for being able to automatically extract dense features, several neural network based approaches have been introduced to perform the end-to-end task of entity extraction and relation extraction. Miwa and Bansal (2016a) introduced an end-to-end neural network model based on syntactic parse tree structures. Given the gold entity mentions, this model gives the current state-of-the-art performance on relation extraction on ACE05 dataset. Katiyar and Cardie (2017) also proposed a joint neural model on entity mention and relation extraction. Their model did not use any syntactic features to achieve comparable performance to Miwa and Bansal (2016a). Their incremental approach to relation extraction is well-suited to also learn entity-level representations and hence we adapt their model for this work.

Zhang et al. (2017b) introduced a position-aware attention to improve slot-filling task. They concatenate word representations with the dense representations for positions relative to the two arguments of the relation being considered in a sentence to learn attention weights for final representation for relation extraction. We adopt their method to include position-aware attention in the model proposed by Katiyar and Cardie (2017).

Recently, Christopoulou et al. (2018) introduced a walk-based model on entity graphs for relation extraction which treats multiple entity-mention pairs in a sentence simultaneously and considers interactions among them. However, their model could not outperform Miwa and Bansal (2016a) on relation extraction on ACE05 dataset and is still at the sentence-level.

Multi-task Learning. Very recent and closest to our approach, Sanh et al. (2019) proposed hierarchical multi-task learning for named entity recognition, entity mention detection, relation extraction and coreference resolution, using labeled data from different sources namely the English portion of OntoNotes 5.0 for named entity recognition and ACE05 corpus for entity mention detection, relation extraction and coreference resolution. Similarly, Luan et al. (2018) introduce a multi-task set-up for identifying and classifying entities, relations, and coreference clusters in scientific articles. They show that their multi-task set-up lead to improvements on all – entities, relations and coreference for the scientific literature. Recently, Luan et al. (2019) introduce a general framework for information extraction using dynamic span graphs. They show that their model can perform entity extraction, coreference resolution and relation extraction all at the same time on several corpora while also making use of an external resource OntoNotes 5.0 for coreference resolution similar to Sanh et al. (2019). Their model shows improvements on several datasets from newswire and scientific literature.

Entity Representations. Recently, a few approaches have been proposed for modeling entities in text. Yang et al. (2016) proposed a reference-aware language models They use coreference to find entity clusters but their entity representation only keeps the most recent entity mention representation restricted to a single token, thus not propagating information about the entity in the document.

Henaff et al. (2016) and Ji et al. (2017) proposed dynamic entity representation. The two models are similar as they dynamically update the entity representations when an entity mention appears in the text. However, Henaff et al. (2016) update all their memory locations for each occurrence of an entity mention thus there are no explicit relation between an entity representation and its memory location. Ji et al. (2017) explicitly model the entities. Thus, each index in their memory represents a discrete entity. For the

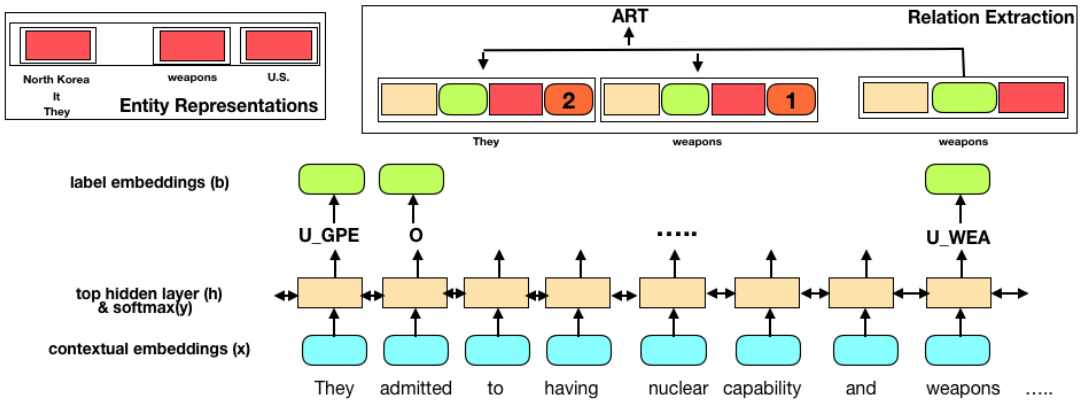


Figure 6.2: Network structure for incrementally learning relations in a sentence shown in Figure 6.1. This snapshot of the network shows entity-level representations from the document and relation extraction for entity "weapons". The top-right part of the figure represents a pointer network over a concatenation of features which include contextual representations, label embeddings, entity-level representations and recency feature. With respect to the current entity mention "weapons", the entity mentions "weapons" in the stack is at position 1 and "They" is at position 2.

entity and relation extraction, we need to model the entities explicitly in order to learn the relations between the entities. Thus, for one of our entity-level representation, we adapt the dynamic entity model Ji et al. (2017) to learn a unique entity representations for each entity cluster in the document.

6.2 Model

Our primary focus for this work is to perform relation extraction under the assumption that the gold entity mentions and gold entities are available. We propose a multi-task setting with entity mention extraction and entity extraction as auxiliary tasks. We will now explain each task in detail.

6.2.1 Entity Mention Extraction

Our entity mention scorer is a standard long short term memory (LSTM) based sequence-to-sequence framework similar to Miwa and Bansal (2016a); Katiyar and Cardie (2017); Zhang et al. (2017a) as shown in Figure 6.2. We label each token in the sentence using the standard BILOU scheme[1]. We compute a score (m_i) for each token (i) in the sentence using hidden representations (h_i). We then compute softmax over the score to learn a probability distribution over all possible tag types as below:

$$P(m_i|m_{<i}) = \text{softmax}(Uh_i) \tag{6.1}$$

We learn to maximize the probability of the correct tag using cross-entropy loss as shown in Figure 6.2.

6.2.2 Entity Extraction

Entity extraction refers to the task of finding for each entity mention its coreferent entity cluster in the document. Unlike the current state-of-the-art coreference resolution models (Lee et al., 2018) which learn a classifier for all pair of entity mentions to learn coreference, we dynamically learn an entity-level representation for each entity cluster as shown in Figure 6.2 and then learn to predict the coreferent entity cluster for each entity mention. These entity-level representations can be assumed to capture the summary of all entity mentions referring to the same entity. We propose different update functions that dynamically update entity-level representations as mentions are encountered in the document from left to right. We use the following three update functions — (1) *average*

[1]Each token is assigned an output tag such as B followed by the entity type for the beginning of an entity mention, I for inside, L for the last token, U for the unit entity mentions and O to represent other non-entity tokens in the sentence.

which averages over the contextual representations of all entity mentions referring to the same entity in the document, (2) *bilinear* function used by Ji et al. (2017) which learn a scalar interpolation factor to update the current entity-level representation with the mention representation and then projects it on to a unit ball after each update, (3) *GRU* which is a simple recurrent unit that takes all the entity mentions in an entity cluster as a sequential input and updates the entity-level representation.

For the task of entity extraction, for each token in the entity mention in the document, our model learns to predict its entity cluster. We employs an attention mechanism that learns to predict the coreferent entity cluster taking their corresponding entity-level representations as input. We will now explain each component in entity extraction.

Entity-level Representations. At the end of the document, we can think of each entity in the document is represented by a dense representation ($\mathbf{e}_j$) which is computed over all the entity mentions ($\mathbf{m}_k$) in their respective coreference cluster (C_j). We experiment with three different functions to update entity representations as we encounter entity mentions in the document.

- **Averaging**: In this approach, we simply average over the dense representations of all the entity mentions in a coreference cluster to obtain entity-level representations. We obtain the dense representation of an entity mention ($\mathbf{m}_k$) by averaging the contextual representation of all the tokens in the entity mention.

$$\mathbf{e}_j = \frac{1}{|C_j|} \sum_k \mathbf{m}_k \tag{6.2}$$

- **Bilinear function**: We use bilinear function to dynamically update entity-level representations as they appear in the document (Ji et al., 2017). All entity-level representations are initialized randomly using a normal distribution ($u \sim \mathcal{N}(0, \sigma^2)$ and then projected onto a unit ball.

$$\mathbf{e}_j = \frac{u}{||u||_2} \tag{6.3}$$

These initial representations are modified as entity mentions appear in the document ($\mathbf{m}_k$) [2] to obtain new entity representations $\mathbf{e}_j$.

$$\delta = \sigma(m_k^T W_\delta \mathbf{e}_j) \tag{6.4}$$

$$\mathbf{u} = \delta \mathbf{e}_j + (1 - \delta) m_k \tag{6.5}$$

$$\mathbf{e}_j = \frac{u}{||u||_2} \tag{6.6}$$

- **Gated Recurrent Unit (GRU)**: In this case, we simply use a GRU over all the coreferent entity mentions in a cluster to obtain its entity-level representations. However these entity mention representations are presented to GRU as they appear in the document and hence there is an order encoded in entity-level representation.

$$\mathbf{e}_j = \text{GRU}(\mathbf{e}_j, \mathbf{m}_k) \tag{6.7}$$

For the task of entity extraction, for each token in the entity mention ($\mathbf{m}_k$) in the document, we learn a classifier which predicts the entity cluster (C_j) it belongs to. At any time step t, let ($\mathbf{m}_k$) be the current token of the entity mention and N entities have occurred until t such that their representations are $\mathbf{e}_1$, $\mathbf{e}_2$, ... $\mathbf{e}_N$. Additionally, we also use mention-level features $\mathbf{m}_1$, $\mathbf{m}_2$, ... $\mathbf{m}_N$ which refers to the most recent entity mentions which modified their corresponding entity-level representations. We learn attention weights corresponding to each entity cluster and a new entity cluster (N+1), which

[2] We still average the top-hidden layer representations of all the tokens in the entity mention to obtain the mention representation.

represents the start of a new entity clusters, as below.

$$u_i = \mathbf{v}^T \tanh(\mathbf{W}_e[\mathbf{e}_i; \mathbf{m}_i] + \mathbf{W}_m \mathbf{m_k}) \tag{6.8}$$

$$a_i = \frac{\exp(u_i)}{\sum_{j=1}^{N+1} \exp(u_j)} \tag{6.9}$$

Here, $\mathbf{W}_e$, $\mathbf{W}_m \in \mathbb{R}^{d_a \times d}$ and $\mathbf{v} \in \mathbb{R}^{d_a}$ are parameters, where d is the dimension of the hidden states and d_a is the size of the attention layer. We train our model to learn to predict highest attention weight to the coreferent entity cluster similar to the idea proposed in Katiyar and Cardie (2017).

6.2.3 Relation Extraction

We predict relations incrementally for each gold entity mention m_k in the sentence. We employ a pointer network (Vinyals et al., 2015a) similar to Katiyar and Cardie (2017) to find the related entity mentions as shown in Figure 6.2. We use position-aware attention model Zhang et al. (2017b) to capture sentential context relative to the entity mentions of interest.

We will now explain the position-aware attention model (Zhang et al., 2017b) used in this work. Let $X_{m_k} = \{x_{m_{k1}}, x_{m_{k1}+1}, \ldots x_{m_{k2}}\}$ is one of the entity mention of interest. The approach learns a position sequence $p_i^{m_k}$ for all the tokens (i) in the sentence relative to the entity mention where:

$$p_i^{m_k} = \begin{cases} i - m_{k1}, & i < m_{k1} \\ 0, & m_{k1} \geq i \leq m_{k2} \\ i - m_{k2}, & i > m_{k2} \end{cases} \tag{6.10}$$

where m_{k1} and m_{k2} are the indices of the first and last token in the entity mention and $p_i^{m_k}$ is the relative distance of each token at position i in the sentence to the entity mention.

We transform these position sequence into position embeddings $\mathbf{p}_i^{m_k}$ using an embedding matrix $\mathbf{P}$. Next, we compute an attention weight a_i for each token i conditioned on a pair of entity mention m_j and m_k as:

$$u_i = \mathbf{v}^T \tanh(\mathbf{W}_h \mathbf{h}_i + \mathbf{W}_q \mathbf{q} + \mathbf{W}_{m_j} \mathbf{p}_i^{m_j} + \mathbf{W}_{m_k} \mathbf{p}_i^{m_k}) \tag{6.11}$$

$$a_i = \frac{\exp(u_i)}{\sum_{j=1}^{N+1} \exp(u_j)} \tag{6.12}$$

where, $\mathbf{W}_h$, $\mathbf{W}_q \in \mathbb{R}^{d_a \times d}$, $\mathbf{W}_{m_j}$, $\mathbf{W}_{m_k} \in \mathbb{R}^{d_a \times d_p}$ and $v \in \mathbb{R}^{d_a}$ are the parameters of the network, where d is the dimension of the hidden states and d_a is the size of the attention layer and d_p is the dimension of the position embeddings.

Next, we compute the sentence representation using these attention weights a_i for each token i in the sentence. Thus, the final representation ($z_{j,k}$) which captures the context relative to the entity mentions m_j and m_k is:

$$z_{j,k} = \sum_{i=1} a_i \mathbf{h}_i \tag{6.13}$$

In addition to the context computed above, we propose two novel features for relation extraction. First, Katiyar and Cardie (2017) use a pointer network to identify relations for each entity mention. They stack all the candidate entity mentions together as shown in Figure 6.2. We instead propose the idea of using the indices ($r_{j,k}$) of the entity mentions in this stack as a measure of *recency* of the candidate entity mentions (m_j) with respect to the current entity mention (m_k). Thus, the most recent or nearest entity mention "weapons" will be at position 1 in the stack and so on. Our idea tries to capture distance between entity mentions where representing distance by the number of tokens between entity mentions is sparse. Second, we use explicit entity-level representations ($\mathbf{e}_j$) and ($\mathbf{e}_j$) as introduced in Section 6.2.2 for the candidate entity mention and the current entity mention respectively. We learn attention weights corresponding to all

the candidate entity mentions m_j, $1 \geq \mathrm{j} < M$ as below:

$$u_j = \mathbf{v}^T \tanh(\mathbf{W}_m[\mathbf{e}_j; \mathbf{r}_{j,k}; \mathbf{z}_{j,k}] + \mathbf{W}_e \mathbf{e}_k) \tag{6.14}$$

$$a_j = \frac{\exp(u_j)}{\sum_{q=1}^{M} \exp(u_q)} \tag{6.15}$$

Here, $\mathbf{W}_m \in \mathbb{R}^{d_a \times d_q}$, $\mathbf{W}_e \in \mathbb{R}^{d_a \times d_p}$ and $\mathbf{v} \in \mathbb{R}^{d_a}$ are parameters, where d is the dimension of the hidden states and d_a is the size of the attention layer and d_q is the sum of the dimension of the recency embedding, the dimension of the sentence representation in Equation 6.13 and the dimension d_p of the entity representation in Equation 6.7.

We train our model similar to Katiyar and Cardie (2017) to predict the highest attention weights between related entity mentions. We also use a threshold to predict multiple relations for an entity mention during test time.

6.3 Training

We train our network by maximizing the log probability of the correct entity mention sequence M, entity cluster assignment E and relations among the entity mentions R in a sentence S. However, our entity mention detection and entity extraction is an auxiliary task, thus we experiment with different weights λ_m, λ_e for this task using the development set.

$$\begin{aligned} \log P(M, E, R|D) \approx & \lambda_m \sum_{i \in |S|} \log P(m_i | m_{g_{i-1}}) \\ & + \lambda_e \sum_{j \in |M_g|} \log P(e_j | M_g, C_{j,g}) \\ & + \sum_{j \in |M_g|} \log P(r_j | M_g, C_{N,g}) \end{aligned}$$

where, M_g and C_g are gold-standard entity mentions and entity coreference clusters. We also experiment with variations of our model to output multiple relations as suggested in Katiyar and Cardie (2017).

At inference time, we use the gold entity mentions and entity coreference clusters to output the relations.

6.4 Experiments

In this section, we provide details about the datasets, evaluation metrics, baselines and previous models, hyperparameters and training details and then present results and discussion of the results.

6.4.1 Data

We evaluate our proposed model on ACE05 dataset from the Automatic Content Extraction program. The dataset is annotated with 7 main entity types – Person (PER), Organization (ORG), Geographical Entities (GPE), Location (LOC), Facility (FAC), Weapon (WEA) and Vehicle (VEH). For each entity, all its entity mentions are annotated within the document. Hence, all these entity mentions for a given entity form the entity cluster. For each entity mention, both the phrase and its head are annotated. In this work, similar to Li and Ji (2014); Miwa and Bansal (2016a); Katiyar and Cardie (2017); Zhang et al. (2017a) we only extract the head of the entity mention. Thus, our annotations for entity mentions are not overlapping. The dataset contains relation annotations for 6 relation types namely – Physical (PHYS), Person-Social (PER-SOC), Organization-Affiliation (ORG-AFF), Agent-Artifact (ART), GPE-Affiliation (GPE-AFF) and Part-

Whole (PART-WHOLE) relation. We use the same dataset split as Li and Ji (2014); Miwa and Bansal (2016a); Katiyar and Cardie (2017); Zhang et al. (2017a). There are 351 documents in the training set, 80 in the development set and the remaining 80 in the test set.

6.4.2 Evaluation Metrics

We report precision, recall and f-score on relation extraction at *mention-level* similar to Miwa and Sasaki (2014); Miwa and Bansal (2016a); Katiyar and Cardie (2017). A relation is considered correct if we can identify the head of the relation arguments and the relation type.

In ACE05 dataset, the relations are limited to entity mentions within a sentence, however there can be several coreferent entity mentions in the same sentence. For example, in Figure6.1 "North Korea" and "it" are coreferent entity mentions. However, the gold annotation only annotates the ARTIFACT relation between the entity mentions "it" and "weapons". Any system that predicts ARTIFACT relation between "North Korea" and "weapons" will be considered incorrect. For this reason, we also evaluate our model on *entity-level* evaluation metric. For computing both precision and recall, we collapse all the mention-level relation predictions to entity-level relation prediction and mention-level gold predictions to entity-level gold predictions and compute them at the entity-level. A relation is now considered correct if we identify the correct entity clusters for the relation arguments and the relation type.

6.4.3 Baselines and Previous Models

We compare our proposed systems with three previous models for relation extraction on ACE05 dataset. Miwa and Bansal (2016a)[3] proposed tree-based model on shortest dependency paths for relation extraction and achieve the current state-of-the art performance on ACE05 dataset. Katiyar and Cardie (2017) proposed a pointer networks based model for relation extraction which does not require any external syntactic features. Christopoulou et al. (2018) proposed an approach which treats multiple relations simultaneously in a sentence and considers interactions among them. However, this approach performed comparably to (Miwa and Bansal, 2016a).

We present different approaches for relation extraction based on Katiyar and Cardie (2017). First, we incorporate only position-aware attention, recency and label embeddings in addition to Katiyar and Cardie (2017) which we refer to as our sentence-level model. Next, we append BERT embeddings to the original word2vec embeddings to the sentence-level model. On this model, we then experiment with the different entity-level representations such as the averaging method (Our model+average), bilinear function (Our model+ bilinear) and GRU (Our model+GRU).

6.4.4 Hyperparameters and Training Details

We use Adam (Kingma and Ba, 2014) for training our network with initial learning rate as 0.001 with exponential decay and gradient clipping to a value of 5. We initialize our word embeddings with 200-dimensional word2vec (Mikolov et al., 2013) embeddings trained on Wikipedia and fine-tune these embeddings during training. We concatenate

[3]Their proposed model was retrained using gold entity mentions to achieve higher performance than the one reported in the paper for relation extraction.

	P	R	F1
Miwa and Bansal (2016a)	70.1	61.2	65.3
Katiyar and Cardie (2017)	59.9	66.9	63.2
Christopoulou et al. (2018)	69.7	59.5	64.2
Sentence-level features			
Our model	73.0	62.4	67.2
Our model + BERT	75.4	70.0	72.7
Sentence+Entity-level features			
+ average	73.4	70.1	71.7
+ bilinear	74.5	70.6	72.5
+ GRU	**77.5**	70.4	**73.8**

Table 6.1: Relation performance on ACE05 test set.

word embeddings with fixed 768-dimensional pretrained BERT embeddings (Devlin et al., 2018) obtained from averaging the top four layers We also employ dropout (Srivastava et al., 2014) on the input and output layer to avoid overfitting. All weights are initialized from small random uniform noise. The dimensionality of the hidden units for our network is 70 and the number of hidden layers are 2.

6.5 Results and Discussion

Sentence-level Features. We show the results of our model using only sentence-level features which include position-aware attention, recency and label embeddings in addition to Katiyar and Cardie (2017) in Table 6.1. Our model outperforms all previous models for relation extraction on ACE05 dataset. We report F-score of 67.2 which is an improvement of 2 points in F1 over the previous state-of-the-art.

We also show that using fixed BERT embeddings our model further improves relation extraction by 5.5 points F1. We obtained these embeddings from pretrained BERT

	P	R	F1
Our model + BERT	78.5	71.9	75.0
+ GRU	80.1	71.4	75.5

Table 6.2: Entity-level relation performance

model (Devlin et al., 2018) on each sentence independently in a document and used them in concatenation with the word2vec vectors. We present state-of-the-art results on relation extraction on ACE05 dataset.

We also perform ablation study as shown in Table 6.3 to find the effect of the *recency* feature. As expected, our model gains significant improvement (+1.5 F1) by utilizing the recency information as features automatically extracted by LSTM do not necessarily capture this information. Thus, we show that our model gains by utilizing the recency feature for relation extraction.

	P	R	F1
Our model + BERT	75.4	70.0	72.7
- Recency	73.4	68.7	71.0

Table 6.3: Ablation study to analyse the effect of recency on relation extraction.

Entity-level Features. We also show the performance of the three update functions we introduced in Section 6.2.3 in Table 6.1. We find that GRU based update function improves the performance on relation extraction by 1 point F1. However, we did not see similar improvements using the average and bilinear update function.

We evaluate our models at entity-level instead for all mentions such that effect of finding multiple instances of the same relation are not reflected in the performance. We show in Table 6.2 that the GRU based entity level representations still improves relation extraction. Thus, our model is improving by not only extracting multiple instances of

	P	R	F1
Our model + GRU	77.5	70.4	73.8
- Entity Representation	76.1	68.8	72.3
- Mention extraction	77.0	68.1	72.2

Table 6.4: Effect of auxiliary tasks on relation extraction.

the same relation but is able to learn other document-level characteristics of relation extraction task and is able utilize it to improve the performance.

Auxiliary Tasks. Next, we also study the effect of introducing mention extraction and entity extraction as auxiliary tasks in our model.

As shown in Table 6.4, we first ablate using explicit entity-level representations in our model. However, our model still performs entity extraction. We find that performance of our model drops by 1.5 points F1. We conclude that only using entity extraction (or coreference resolution) as the auxiliary task does not improve relation extraction. Moreover, our model performs slightly worse than even our relation extraction model with only sentence-level features as shown in Table 6.1. This could potentially indicate that the improvement in relation extraction as shown by Sanh et al. (2019); Luan et al. (2018, 2019) using mention extraction and coreference resolution in a multi-task setting is a consequence of coreference resolution improving mention extraction. However, in this work, we show that aggregating information over all the mentions in a coreference chain and using them explicitly is important for improving relation extraction. We also modeled mention extraction as another auxiliary task but do not find significant effects on relation extraction suggesting that in the event of the availability of gold entity mentions, the auxiliary task of mention extraction does not improve relation extraction.

Predicted Coreference. We also evaluate our model when gold coreference chains are not available. We use predictions from a state-of-the-art coreference system (Lee et al., 2018). We prune these predictions based on the gold entity mentions. Hence, we do not introduce any error based on the incorrect entity mention prediction by the coreference system. Most of the current coreference systems often predict mentions and coreference relations together, thus we resorted to this simple pruning strategy. We learn entity-level representations by aggregating over all the mentions in the predicted coreference chains. We find that the performance of Our model + GRU drops to an F1 score of 72.2; this drop is expected because the coreference model is trained on a different dataset (OntoNotes) and using the final prediction directly from such a system without fine-tuning on the current dataset is brittle. This result however also motivates us in future work to incorporate coreference resolution into our model and to learn entity-level representations without requiring gold-standard coreference information.

6.6 Analysis

We perform analysis on the entity-level representations that we learn using the average, bilinear and GRU function. We hypothesized that our entity-level representation might be able to learn the importance of an entity in a document and thus encode that such entities are more likely to be related to other entities. We will refer to these entities as salient entities.

In the domain of Information Retrieval, salient entities Gillick and Dunietz (2014) are, in an empirical definition, those that human readers deem most relevant to the document. Most often, some of the features that are found to be useful for determining salient entities include index of the sentence in which the first mention of the entity appears,

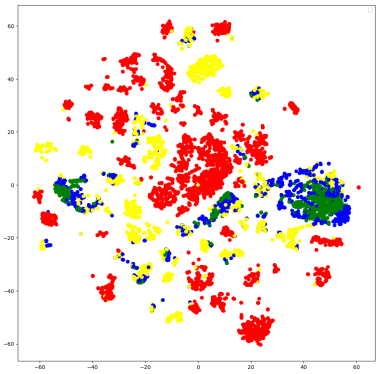


Figure 6.3: Average entity-level representations

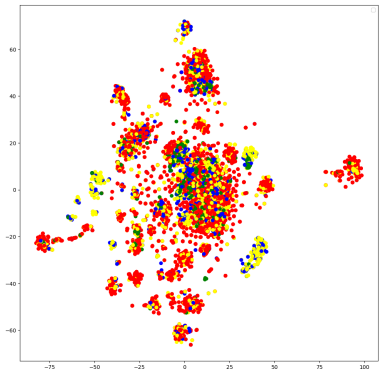


Figure 6.4: Bilinear entity-level representations

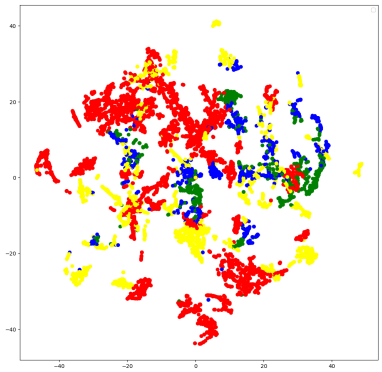


Figure 6.5: GRU entity-level representation

Figure 6.6: Figure showing the t-SNE plots for different types of entity-level representations.

number of times the head word of the entitys first mention appears, conjunction of the number of total mentions of the entity. Inspired by these features, we perform analysis of our entity representations based on the frequency counts of the entity mentions in a document. Thus, we would like to know if the entity representations that we learnt via average, bilinear and GRU are predictive of the frequency of the entity in a document.

Visualization. We first project our learnt 70-dimensional entity-level representations into 2-dimensional representation using t-SNE (van der Maaten and Hinton, 2008) for average, bilinear model and GRU model as shown in Figure 6.3, Figure 6.4 and Figure 6.5 respectively. We then bin these representation based on the frequency of entity mentions and show them in different colors. In Figure 6.3, Figure 6.4 and Figure 6.5, red represents entities with frequency [1], yellow with frequency in range [2, 4], blue in range [5, 7] and green in [8, above].

Clearly, we see that when we learn entity-level representations using GRU, then these embeddings can be differentiated based on the frequency of the entities whereas when we learn them via bilinear function which includes projecting them into unit ball then these representations lose the frequency information and all the entity representations for different frequencies are merged together. We also find that entity-level representations when learned using the average function also retain frequencies of the entities as the different colors can be easily distinguished and can be seen forming different clusters except for the blue and green colors. These colors represent the entities with frequencies greater than 5 thus showing that learning representations with the average function looses frequency information for entities with higher frequencies.

Classification. We also learn a binary classifier given the 70-dimensional entity representation as features to predict whether an example belongs to low frequency (count ≤ 4) entity or a high frequency (count > 4) entity. We randomly split the examples into training (80%) and test (20%). We show the performance on this task in Table 6.5. We find that the GRU entity-level representation are extremely good at predicting the frequency of an entity, whereas Bilinear model performs even worse than a majority baseline. Thus, we show again that the bilinear model does not capture frequency of the entity. As we also discussed above, the Average model is also able to capture the

Method	Accuracy
Majority-baseline	0.785
Average	0.960
Bilinear	0.540
GRU	0.974

Table 6.5: Classification accuracy for different methods of entity-level representations.

frequency of the entity mentions but not as efficiently as the GRU model and hence it performs worse than the GRU model on this prediction task which is an average over 5-fold classification.

In summary, in this section we show that our three proposed ways of learning entity-level representations capture the frequencies of the entities to different extent. We found that the entity representations when learned using the GRU function improves the performance on relation extraction. Thus, the capability of this GRU function to capture frequencies is one possible explanation for the improvements shown in relation extraction. However, this is definitely not the only explanation and in future, we want to understand these entity-level representations better in terms of other interpretable explanations.

6.7 Chapter Summary

We proposed using novel sentence-level features in the form of recency of entity mentions and document-level features in the form of entity-level representations to the existing models for relation extraction. We show that both these features significantly improve relation extraction and achieve state-of-the-art performance. We also show that modeling coreference resolution only as an auxiliary task is not sufficient to improve relation extraction even under the assumption of the availability of gold mentions. Thus,

we need to learn entity-level representations to aggregate the information across different mentions of the same entity in the document and use these as features for the task. Our proposed model depends on the availability of gold coreference at test time, however, it will be interesting to learn entity-level representations without requiring gold coreference information during prediction.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this dissertation, we propose neural network based models for information extraction for learning facts and opinions from text. In particular, we show that without the need of feature-engineering, our deep learning models are able to produce state-of-the-art results in information extraction. We propose recurrent neural network based models which can learn entities and relations jointly, can also learn nested named entities which have often been ignored in the past owing to its complexity of the output space. We also forward the research in relation extraction by extending it to include features from the entire document instead of only the sentence and thus moving a step forward to realising the goal of information extraction to perform entity mention extraction, coreference resolution, entity mention classification and relation extraction all at the same time.

7.1 Summary of Contributions

In Chapter 3, we propose a LSTM-based model for the task of fine-grained opinion mining. We extract the opinion entities and relation jointly with our approach. We propose a very simple joint sequential output representation for opinion entities and opinion relations such as the opinion entities are represented using a standard tagging scheme, however our relation labels are represented by the distance between the entity mentions in terms of the number of tokens in between them in the sentence. We compare our model with a feature-engineered state-of-the-art model that extracts entities and relations in a pipelined manner and later performs joint inference on both entities and relations based on a set of constraints. This model uses an extensive list of features including opinion lexicons, semantic and syntactic features such as dependency trees, etc. We show that

our simple model with a simple representation can perform within 1-3% on both opinion entities and opinion relation extraction.

Our proposed idea of representing relations with only the distance between the entity mentions was very successful in extracting opinion entities and relations. However, the task of extracting facts entails extracting entities and relations in a more general setting where entities are often related to multiple other entities. Thus, for our model to be more general, we propose extracting relations with Pointer networks as explained in Chapter 4. We compare our model with the then state-of-the-art neural network based model which uses shortest dependency path between the entity mentions to extract relations. In chapter 4, we show that our model which does not depend on the availability of dependency paths instead uses attention over entity mentions to learn relations between them performs comparably to the previous model. Thus, we propose a model for joint entity and relation extraction which does not depend on external syntactic features.

As we focus on extracting entities and relations, we often make an assumption that the entity mentions are not overlapping or nested. For this reason, in opinion mining all approaches perform preprocessing of the datasets to remove any overlapping entity mentions. Similarly, for the general task of information extraction, the approaches focus on extracting only the head phrase of the entity mentions which as a result are not overlapping. In Chapter 5, we focus on developing models for nested entity mentions by using a hypergraph output representation. We show in chapter 5 that we can learn this hypergraph structure efficiently in a greedy fashion to extraction both nested entity mentions as well as their heads. We outperform previous feature-based approaches by a significant margin on both news wire as well as biomedical domain. This also shows that these recurrent neural are capable of learning complex structures and can outperform previous feature-based approaches where generating a list of features for nested named

entities task is challenging.

Most of the work we discussed so far deals with extracting entity mentions and relations using sentence-level context. Thus, relations are only extracted with respect to the entity mentions which means that that the relation tuples often contain pronominal mentions of entities without referring to the entity itself. Thus, there is an important missing component to information extraction which involves identifying coreference relations between the entity mentions. To this effect, we propose to learning entity-level information in chapter 6 which captures the summary of all mentions of an entity. In our experiments, we show that using gold coreference chains to learn these entity representations and using them as additional features in relation extraction, our models outperform the previous state-of-the-art results. Thus, we propose to use document-level context into our models and show improvements in relation extraction showing promise towards document-level information extraction.

7.2 Future Work

In this section, we will discuss some of the future directions that can help build better neural models for information extraction and downstream tasks such as question answering or dialogue systems.

Interpretation of neural networks. Most of the neural network based models described in this thesis are based on LSTMs, a variant of recurrent neural networks. These models apply recurrent non-linear gated operations on vectors as explained in chapter 3. These transformations help these networks to learn complex dense representations suitable for learning several syntactic and semantic tasks in natural language processing.

However, this complexity also leads to these networks being uninterpretable unlike the previous feature-based models which learned weights for pre-decided list of discrete features and thus more interpretable. The uninterpretability of neural networks is limiting for research because it is difficult to suggest improvements for these model based on their errors on examples. The only handle that we currently have is better tuning of hyperparameters which can lead to the improvement in performance of the model. Thus, we think that it is extremely important to understand these neural network models both in terms of the features they are learning as well as their parameter space. Understanding these neural networks can lead to building better models for tasks in natural language processing. Also, interpretability of neural networks can help in the explainability of these models thus expanding their applications to problems which require explainable models such as in medicine, etc.

Multi-task Learning. As we concluded in this thesis, tasks such as entity extraction and coreference resolution helps in improving the performance of relation extraction. Thus, there are many semantically related tasks which if modeled jointly can lead to the improvement of all the tasks. However, there are challenges that lie ahead when modeling different tasks together in a neural network setting. For example, in chapter 6 we show that learning entity-level representation using gold coreference improves relation extraction. However, we want to be able to use predicted coreference to learn these entity-level representations as gold coreference annotations are difficult to obtain. Though using predicted coreference can potentially lead to cascading errors and most of the datasets are not annotated with gold coreference. Thus, it is a challenge to train the system end-to-end to avoid cascading error.

More so, we explained in the introduction, that learning these structured information can potentially help downstream tasks such as question answering as shown in Fig-

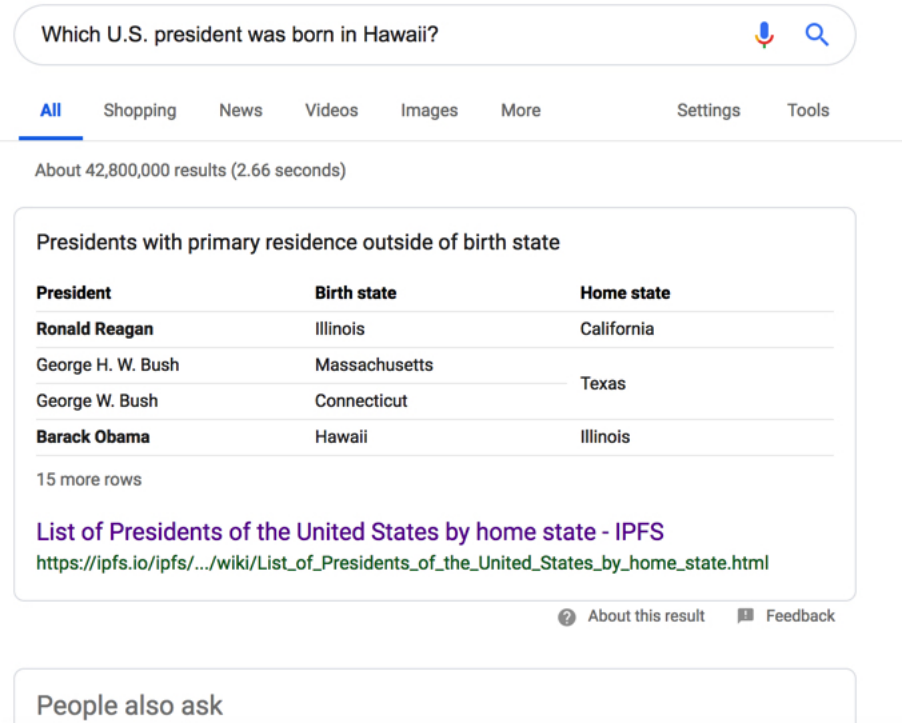


Figure 7.1: Figure showing an example of search result from the popular search engine Google.

ure 1.2. Recently, there has been a splurge of models in reading comprehension(Chen, 2018) which perform end-to-end question answering without any intermediate representation. These systems are employed in search engines to answer questions from the users as shown in Figure 1.2. These systems are able to correctly answer "Which U.S. president was born in Honolulu?" However, when these systems are asked to answer question such as "Which U.S. president was born in Hawaii?" as shown in Figure 7.1, these systems are not able to extract "Barack Obama" as the answer. This goes on to show that these extractive systems can use relation information between entities "Honolulu" and "Hawaii" to answer both these questions. Thus, we need to think about the intermediate representation of the structured knowledge which can be used by these question answering systems to perform reasoning over them and answer these questions.

Learning from less labeled examples. In this thesis, we mostly focused on structured prediction problems which require annotation at the token level. These datasets can be extremely tedious to develop and as a result there are only a few datasets that are available for information extraction. However, with the introduction of pretrained word embeddings like word2vec which capture semantic similarity of words, the models can get some prior knowledge about words and thus the models can get benefit from them. Recently, with contextual word embeddings such as BERT(Devlin et al., 2018), these embeddings are pretrained on a very large corpus using language modeling. Thus, these embeddings can also potential capture more complex relationships of words with each other in a sentence in addition to the semantics of the words. When we used these embeddings for relation extraction in chapter 6, we see significant improvement of 5 absolute points in F1 score. Thus these pretrained embeddings capture semantics of words and their syntactic relations with reach other which paves the path to learning relations with less labeled data. We want to be able to learn relations with only few labeled examples without the need to annotate hundreds of examples for each relation type.

BIBLIOGRAPHY

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, BioNLP '07, pages 65–72. http://dl.acm.org/citation.cfm?id=1572392.1572404.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. ICLR*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *CoRR* abs/1506.03099. http://arxiv.org/abs/1506.03099.

Yoshua Bengio. 2009. Learning deep architectures for ai. *Found. Trends Mach. Learn.* 2(1):1–127. https://doi.org/10.1561/2200000006.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, New York, NY, USA, ICML '09, pages 41–48. https://doi.org/10.1145/1553374.1553380.

Rishi Bommasani, Arzoo Katiyar, and Claire Cardie. 2019. SPARSE: Structured prediction using argument-relative structured encoding. In *Proceedings of the Third Workshop on Structured Prediction for NLP*. Association for Computational Linguistics, Minneapolis, Minnesota, pages 13–17. https://www.aclweb.org/anthology/W19-1503.

Eric Breck, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of the 20th International Joint Conference on Artifical*

Intelligence. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'07, pages 2683–2688. http://dl.acm.org/citation.cfm?id=1625275.1625707.

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '05, pages 724–731. https://doi.org/10.3115/1220575.1220666.

Yee Seng Chan and Dan Roth. 2010. Exploiting Background Knowledge for Relation Extraction. In *Proc. of the International Conference on Computational Linguistics (COLING)*. Beijing, China. http://cogcomp.org/papers/ChanRo10.pdf.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '11, pages 551–560. http://dl.acm.org/citation.cfm?id=2002472.2002542.

Kai-Wei Chang, Rajhans Samdani, and Dan Roth. 2013. A constrained latent variable model for coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 601–612. http://www.aclweb.org/anthology/D13-1057.

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceed-*

ings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers. http://aclweb.org/anthology/P/P16/P16-1223.pdf.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *SIGKDD Explor. Newsl.* 19(2):25–35. https://doi.org/10.1145/3166054.3166058.

Xilun Chen, Arzoo Katiyar, Xiaoan Yan, Lu Wang, Carmen Banea, Yoonjung Choi, Lingjia Deng, Claire Cardie, Rada Mihalcea, and Janyce Wiebe. 2014. Cornpittmich sentiment slot-filling system at TAC 2014. In *Proceedings of Text Analysis Conference, TAC 2014*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 484–494. http://www.aclweb.org/anthology/P16-1046.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Sydney, Australia, pages 431–439. http://www.aclweb.org/anthology/W/W06/W06-1651.

Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for

Computational Linguistics, Stroudsburg, PA, USA, HLT '05, pages 355–362. https://doi.org/10.3115/1220575.1220620.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2018. A walk-based model on entity graphs for relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 81–88. http://aclweb.org/anthology/P18-2014.

Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Comput. Linguist.* 31(1):25–70. https://doi.org/10.1162/0891201053630273.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*. ACM, New York, NY, USA, ICML '08, pages 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* 12:2493–2537. http://dl.acm.org/citation.cfm?id=1953048.2078186.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* .

Cícero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. *CoRR* abs/1504.06580. http://arxiv.org/abs/1504.06580.

Jeffrey L. Elman. 1990. Finding structure in time. *COGNITIVE SCIENCE* 14(2):179–211.

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Singapore, pages 141–150. http://www.aclweb.org/anthology/D/D09/D09-1015.

R Florian, H Hassan, A Ittycheriah, H Jing, N Kambhatla, X Luo, N Nicolov, and S Roukos. 2004. A statistical model for multilingual entity detection and tracking. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*. Association for Computational Linguistics, Boston, Massachusetts, USA, pages 1–8.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Comput. Linguist.* 28(3):245–288. https://doi.org/10.1162/089120102760275983.

Dan Gillick and Jesse Dunietz. 2014. A new entity salience task with millions of training examples. In *Proceedings of the European Association for Computational Linguistics*.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*. pages 273–278. https://doi.org/10.1109/ASRU.2013.6707742.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*. pages 2537–2547. http://aclweb.org/anthology/C/C16/C16-1239.pdf.

James Hammerton. 2003. Named entity recognition with long short-term memory. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-*

NAACL 2003 - Volume 4. Association for Computational Linguistics, Stroudsburg, PA, USA, CONLL '03, pages 172–175. https://doi.org/10.3115/1119176.1119202.

Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. 2015. Task-oriented learning of word embeddings for semantic relation classification. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Beijing, China, pages 268–278. http://www.aclweb.org/anthology/K15-1027.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 473–483.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2016. Tracking the world state with recurrent entity networks. *CoRR* abs/1612.03969. http://arxiv.org/abs/1612.03969.

Michiel Hermans and Benjamin Schrauwen. 2013. Training and analysing deep recurrent neural networks. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*. pages 190–198. http://papers.nips.cc/paper/5166-training-and-analysing-deep-recurrent-neural-networks.

Salah El Hihi and Yoshua Bengio. 1996. Hierarchical recurrent neural networks for long-term dependencies.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, Franciska de Jong, and Emiel Caron. 2016. A survey of event extraction methods from text for decision support systems. *Decis. Support Syst.* 85(C):12–22. https://doi.org/10.1016/j.dss.2016.02.006.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR* abs/1508.01991. http://arxiv.org/abs/1508.01991.

Ozan Irsoy and Claire Cardie. 2013. Bidirectional recursive neural networks for token-level labeling with structure. *arXiv preprint arXiv:1312.0493* .

Ozan İrsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 720–728. http://aclweb.org/anthology/D14-1080.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 1830–1839. https://www.aclweb.org/anthology/D17-1195.

Rohit J. Kate and Raymond J. Mooney. 2010. Joint entity and relation extraction using card-pyramid parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Stroudsburg, PA, USA, CoNLL '10, pages 203–212. http://dl.acm.org/citation.cfm?id=1870568.1870592.

Arzoo Katiyar and Claire Cardie. 2016. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Association for Computational Linguistics, Berlin, Germany, pages 919–929. https://doi.org/10.18653/v1/P16-1087.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 917–928. https://doi.org/10.18653/v1/P17-1085.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 861–871. https://doi.org/10.18653/v1/N18-1079.

Soo-Min Kim and Eduard Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*. Association for Computational Linguistics, Stroudsburg, PA, USA, SST '06, pages 1–8. http://dl.acm.org/citation.cfm?id=1654641.1654642.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980. http://dblp.uni-trier.de/db/journals/corr/corr1412.htmlKingmaB14.

Dan Klein and Christopher D. Manning. 2001. Parsing and hypergraphs. In *Proceedings of the Seventh International Workshop on Parsing Technologies (IWPT-2001), 17-19 October 2001, Beijing, China*.

Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the 2007*

Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 1249–1258. https://www.aclweb.org/anthology/E17-1117.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 1426–1436. https://www.aclweb.org/anthology/P18-1132.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ICML '01, pages 282–289. http://dl.acm.org/citation.cfm?id=645530.655813.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pages 260–270.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolu-

tion with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 687–692. https://doi.org/10.18653/v1/N18-2108.

Kenton Lee, Tom Kwiatkowski, Ankur P. Parikh, and Dipanjan Das. 2016. Learning recurrent span representations for extractive question answering. *CoRR* abs/1611.01436. http://arxiv.org/abs/1611.01436.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*. pages 402–412. http://aclweb.org/anthology/P/P14/P14-1038.pdf.

Bing Liu and Lei Zhang. 2012. *A Survey of Opinion Mining and Sentiment Analysis*, Springer US, Boston, MA, pages 415–463. https://doi.org/10.1007/978-1-4614-3223-$4_1$3.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1433–1443. https://aclweb.org/anthology/D/D15/D15-1168.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 857–867. http://aclweb.org/anthology/D15-1102.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 3219–3232. http://aclweb.org/anthology/D18-1360.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. *CoRR* abs/1904.03296. http://arxiv.org/abs/1904.03296.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 879–888. https://aclweb.org/anthology/D/D15/D15-1104.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Lisbon, Portugal, pages 1412–1421. http://aclweb.org/anthology/D15-1166.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. 2008. Semantic role labeling: An introduction to the special issue. *Comput. Linguist.* 34(2):145–159. https://doi.org/10.1162/coli.2008.34.2.145.

André F. T. Martins and Ramón F. Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of the 33rd International Conference on International Conference*

on Machine Learning - Volume 48. JMLR.org, ICML'16, pages 1614–1623. http://dl.acm.org/citation.cfm?id=3045390.3045561.

Ryan McDonald and Fernando Pereira. 2005. Identifying gene and protein mentions in text using conditional random fields. *BMC Bioinformatics* 6(1):S6. https://doi.org/10.1186/1471-2105-6-S1-S6.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pages 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics, Suntec, Singapore, pages 1003–1011. http://www.aclweb.org/anthology/P/P09/P09-1113.

Makoto Miwa and Mohit Bansal. 2016a. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 1105–1116.

Makoto Miwa and Mohit Bansal. 2016b. End-to-end relation extraction using lstms on sequences and tree structures. *CoRR* abs/1601.00770. http://arxiv.org/abs/1601.00770.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical Meth-*

ods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. pages 1858–1869. http://aclweb.org/anthology/D/D14/D14-1200.pdf.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 2598–2608. https://www.aclweb.org/anthology/D17-1275.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes* 30.

Ramesh Nallapati, Bing Xiang, and Bowen Zhou. 2016. Sequence-to-sequence rnns for text summarization. *CoRR* abs/1602.06023. http://arxiv.org/abs/1602.06023.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. Association for Computational Linguistics, pages 39–48. https://doi.org/10.3115/v1/W15-1506.

Vlad Niculae, Kai Sun, Xilun Chen, Yao Cheng, Xinya Du, Esin Durmus, Arzoo Katiyar, and Claire Cardie. 2016. Cornell belief and sentiment system at TAC 2016. In *Proceedings of the 2016 Text Analysis Conference, TAC 2016, Gaithersburg, Maryland, USA, November 14-15, 2016*.

Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. 2002. The genia corpus: An annotated research abstract corpus in molecular biology domain. In *Proceedings of the Second International Conference on Human Language Technology Research*. Mor-

gan Kaufmann Publishers Inc., San Francisco, CA, USA, HLT '02, pages 82–86. http://dl.acm.org/citation.cfm?id=1289189.1289260.

Mary Ellen Okurowski. 1993. Information extraction overview. In *Proceedings of the TIPSTER Text Program: Phase I*. Association for Computational Linguistics, Fredericksburg, Virginia, USA, pages 117–121.

Joonsuk Park, Arzoo Katiyar, and Bishan Yang. 2015. Conditional random fields for identifying appropriate types of support for propositions in online user comments. In *Proceedings of the 2nd Workshop on Argumentation Mining*. Association for Computational Linguistics, Denver, CO, pages 39–44. https://doi.org/10.3115/v1/W15-0506.

Sachin Pawar, Girish K. Palshikar, and Pushpak Bhattacharyya. 2017. Relation extraction : A survey. *CoRR* abs/1712.05191. http://arxiv.org/abs/1712.05191.

S Pyysalo, F Ginter, H Moen, T Salakoski, and S Ananiadou. 2013. *Distributional Semantics Resources for Biomedical Text Processing*, pages 39–44.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*. https://www.aclweb.org/anthology/W95-0107.

Josef Ruppenhofer, Michael Ellsworth, Miriam R.L. Petruck, Christopher R. Johnson, and Jan Scheffczyk. 2006. *FrameNet II: Extended Theory and Practice*. International Computer Science Institute, Berkeley, California. Distributed with the FrameNet data.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. *AAAI* .

Sunita Sarawagi and William W Cohen. 2005. Semi-markov conditional random fields for information extraction. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, MIT Press, pages

1185–1192. http://papers.nips.cc/paper/2648-semi-markov-conditional-random-fields-for-information-extraction.pdf.

Jürgen Schmidhuber. 1992. Learning complex, extended sequences using the principle of history compression. *Neural Comput.* 4(2):234–242. https://doi.org/10.1162/neco.1992.4.2.234.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proc.* 45(11):2673–2681. https://doi.org/10.1109/78.650093.

Sameer Singh, Sebastian Riedel, Brian Martin, Jiaping Zheng, and Andrew McCallum. 2013. Joint inference of entities, relations, and coreference. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*. ACM, New York, NY, USA, AKBC '13, pages 1–6. https://doi.org/10.1145/2509558.2509559.

Sonit Singh. 2018. Natural language processing for information extraction. *CoRR* abs/1807.02383. http://arxiv.org/abs/1807.02383.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP-CoNLL '12, pages 1201–1211. http://dl.acm.org/citation.cfm?id=2390948.2391084.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15:1929–1958. http://jmlr.org/papers/v15/srivastava14a.html.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. pages 3104–3112. http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9:2579–2605. http://www.jmlr.org/papers/v9/vandermaaten08a.html.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015a. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. pages 2692–2700. http://papers.nips.cc/paper/5866-pointer-networks.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015b. Grammar as a foreign language. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. MIT Press, Cambridge, MA, USA, NIPS'15, pages 2773–2781. http://dl.acm.org/citation.cfm?id=2969442.2969550.

Janyce Wiebe and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. language resources and evaluation. In *Language Resources and Evaluation (formerly Computers and the Humanities*. page 2005.

Craig A. Will. 1993. Comparing human and machine performance for natural language information extraction: Results from the tipster text evaluation. In *Proceedings of the TIPSTER Text Program: Phase I*. Association for Computational Linguistics, Fredericksburg, Virginia, USA, pages 179–193.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '05, pages 347–354. https://doi.org/10.3115/1220575.1220619.

Theresa Ann Wilson. 2008. *Fine-grained Subjectivity and Sentiment Analysis: Recognizing the intensity, polarity, and attitudes of private states*. Ph.D. thesis, The University of Pittsburgh. http://d-scholarship.pitt.edu/7563/.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015a. Semantic relation classification via convolutional neural networks with simple negative sampling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 536–540. http://aclweb.org/anthology/D15-1062.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015b. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1785–1794. http://aclweb.org/anthology/D15-1206.

Bishan Yang and Claire Cardie. 2012. Extracting opinion expressions with semi-markov conditional random fields. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP-CoNLL '12, pages 1335–1345. http://dl.acm.org/citation.cfm?id=2390948.2391100.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Lin-*

guistics (Volume 1: Long Papers). Association for Computational Linguistics, Sofia, Bulgaria, pages 1640–1649. http://www.aclweb.org/anthology/P13-1161.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2016. Reference-aware language models. *CoRR* abs/1611.01628. http://arxiv.org/abs/1611.01628.

Wen-Tau Yih and Dan Roth. 2007. Global inference for entity and relation identification via a linear programming formulation. In L. Getoor and B. Taskar, editors, *An Introduction to Statistical Relational Learning*, MIT Press.

Xiaofeng Yu and Wai Lam. 2010. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics, Stroudsburg, PA, USA, COLING '10, pages 1399–1407. http://dl.acm.org/citation.cfm?id=1944566.1944726.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR* abs/1212.5701. http://arxiv.org/abs/1212.5701.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.* 3:1083–1106. http://dl.acm.org/citation.cfm?id=944919.944964.

Feifei Zhai, Saloni Potdar, Bing Xiang, and Bowen Zhou. 2017. Neural models for sequence chunking. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*. pages 3365–3371.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2017a. End-to-end neural relation extraction with global optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1730–1740. http://aclweb.org/anthology/D17-1182.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017b. Position-aware attention and supervised data improve slot filling. In *Empirical Methods in Natural Language Processing (EMNLP)*. https://nlp.stanford.edu/pubs/zhang2017tacred.pdf.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '02, pages 473–480. https://doi.org/10.3115/1073083.1073163.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 427–434. https://doi.org/10.3115/1219840.1219893.